Algorithmes et Protocoles Cryptographiques

Chapitre 2 : Chiffrement symétrique

Mme. Arbia RIAHI
M. Mahmoud TOUNSI

INDP2
SUPCOM, 2022

Plan

- Introduction
- Chiffrements classiques
- Algorithme DES
- Algorithme AES
- Modes de chiffrement

Principe général

- La connaissance de la méthode et de la clé de chiffrement et celle de la méthode et de la clé de déchiffrement **se déduisent facilement l'une de l'autre**.
- Les deux méthodes et les clés sont connues de l'émetteur et du destinataire.

=> L'émetteur et le destinataire doivent se mettre préalablement d'accord sur un secret (la clé) pour utiliser le chiffrement.

- Deux problèmes:
  - L'échange préalable à toute communication sécurisée d'un secret (« la distribution de clés »).
  - Dans un réseau de N entités susceptibles de communiquer secrètement il faut distribuer N*(N-1)/2 clés.

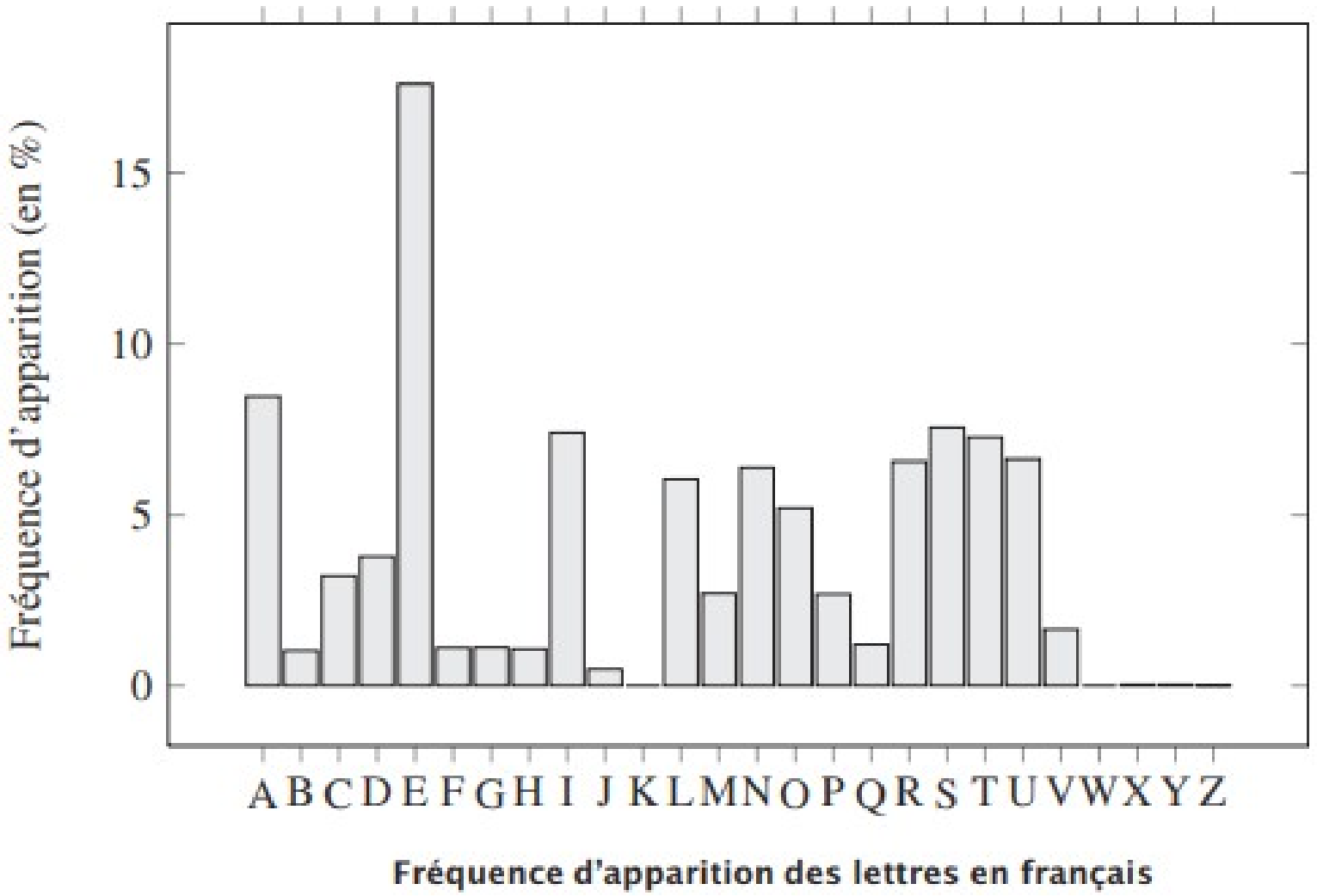
Les méthodes de chiffrement par substitution

- **Principe général:** A chaque lettre ou groupe de lettres on substitue une autre lettre ou groupe de lettres.
- La substitution simple (substitution mono alphabétique): Pour chaque lettre de l'alphabet de base on se donne une autre lettre utilisée dans le texte chiffré.
- Exemple historique: Le chiffrement de César, on décale les lettres de 3 positions :
  - A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
  - D E F G H I J K L M N O P Q R S T U V W X Y Z A B C

Les techniques d'attaque statistique

- Analyse statistique des textes chiffrés.
- Détermination des fréquences d'apparition des symboles.
- Comparaison avec fréquences types caractéristiques des langues.
- Exemple : Fréquences d'apparition (en anglais)
  - Lettres: E 13.05, T 9.02
  - Digrammes: TH 3.16, IN 1.54
  - Trigrammes: THE 4.72, ING 1.42
- Une analyse statistique d'un texte suffisamment long permet de casser un code mono ou même poly-alphabétique
- Le problème est de disposer : de puissance de calcul et du texte suffisant long en regard de la longueur des clés utilisées.

Fréquences d'apparition (Français)



Fréquence d'apparition des lettres en français

La substitution poly-alphabétique

- On utilise une suite de chiffrements mono alphabétiques.
- La suite des chiffrements mono alphabétiques est réutilisée périodiquement.
- Exemple : le chiffrement de Vigenère
  - On prend les 26 chiffrements de César.
  - Les chiffrements associés aux 26 décalages possibles sont représentés par une lettre.
  - Ex : chiffrement avec décalage de k associé à la $k^{ème}$ lettre de l'alphabet.

La substitution poly-alphabétique

- A->B C D E F G H I J K L M N O P Q R S T U V W X Y Z A
- B->C D E F G H I J K L M N O P Q R S T U V W X Y Z A B

- On choisit une clé de répétition comme une suite de lettres : un mot ou une phrase ou un livre.
- Cette clé répétée indéfiniment vis à vis de chaque lettre d'un texte à chiffrer sert à déterminer le chiffrement à utiliser.

Chiffrement de Vigenère (1)

L'outil indispensable : " La table de Vigenère "

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
A	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
B	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A
C	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B
D	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C
E	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D
F	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E
G	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F
H	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G
I	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H
J	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I
K	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J
L	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K
M	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L
N	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M
O	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N
P	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
Q	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
R	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
S	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
T	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
U	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
V	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
W	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
X	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
Y	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
Z	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y

Chiffrement de Vigenère (2)

- On choisit une clé (mot de longueur arbitraire).
- On écrit cette clé sous le message à chiffrer, en la répétant aussi souvent que nécessaire pour que sous chaque lettre du message à chiffrer, on trouve une lettre de la clé.
- Pour chiffrer, on regarde dans le tableau l'intersection de la ligne de la lettre à chiffrer avec la colonne de la lettre de la clé.

Exemple : On veut chiffrer le texte "CRYPTOGRAPHIE" avec la clé "MATHWEB". On écrit la clé sous le texte à chiffrer :

C	R	Y	P	T	O	G	R	A	P	H	I	E
M	A	T	H	W	E	B	M	A	T	H	W	E

Chiffrement de Vigenère (3)

Chiffrement :

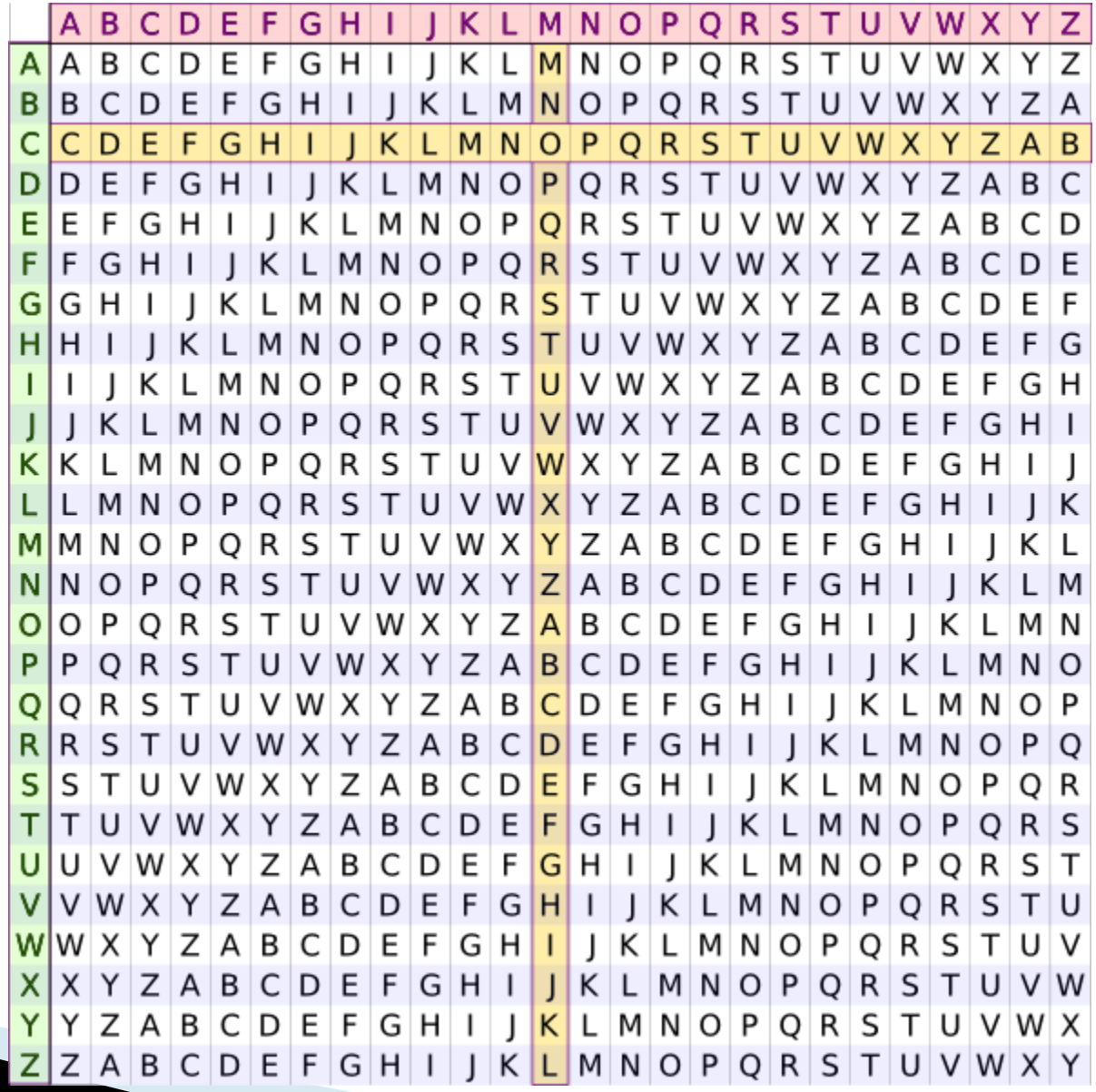
- Pour chaque lettre en clair, on sélectionne la colonne correspondante.
- Pour une lettre de la clé on sélectionne la ligne adéquate.
- Au croisement de la ligne et de la colonne on trouve la lettre chiffrée.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
A	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
B	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A
C	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B
D	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C
E	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D
F	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E
G	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F
H	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G
I	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H
J	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I
K	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J
L	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K
M	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L
N	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M
O	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N
P	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
Q	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
R	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
S	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
T	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
U	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
V	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
W	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
X	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
Y	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
Z	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y

Chiffrement de Vigenère (4)

Le message chiffré :
ORRWPSHDAIOEI

Déchiffrement :
Sur la colonne de la lettre de la clé, rechercher la lettre du message chiffré**.**



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
A	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
B	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A
C	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B
D	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C
E	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D
F	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E
G	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F
H	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G
I	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H
J	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I
K	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J
L	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K
M	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L
N	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M
O	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N
P	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
Q	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
R	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
S	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
T	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
U	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
V	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
W	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
X	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
Y	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
Z	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y

Autres substitutions

- Les substitutions homophoniques : Au lieu d'associer un seul caractère chiffré à un caractère en clair on dispose d'un ensemble de possibilités de substitution de caractères dans laquelle on choisit aléatoirement.
- Les substitutions de polygrammes : Au lieu de substituer des caractères on substitue par exemple des digrammes (groupes de deux caractères)
  - Au moyen d'une table (système de Playfair).
  - Au moyen d'une transformation mathématique (système de Hill).

Système de Playfair (1)

- matrice 5x5 construite en utilisant la clé secrète.
- remplie par les lettres de la clé, déduction faite des doubles (exemple : informatique devient informatque), de gauche à droite et de haut en bas.
- Les cases restantes de la matrice sont remplies avec les lettres restantes dans l'ordre alphabétique (I et J comptent pour une seule lettre).
- Exemple (matrice Playfair avec la clé informatique) :

i	n	f	o	r
m	a	t	q	u
e	b	c	d	g
h	k	l	p	s
v	w	x	y	z

i	n	f	o	r
m	a	t	q	u
e	b	c	d	g
h	k	l	p	s
v	w	x	y	z

Système de Playfair (2)

- Le message en clair est chiffré par groupes de 2 lettres :
  - lettres identiques appartenant à la même paire seront séparées en ajoutant une lettre de remplissage. Si le nombre de lettres est impair, on complète par une lettre de remplissage (ex. ballon → ba lx lo nx).
  - 2 lettres qui se trouvent dans la même ligne de la matrice seront décalées d'une case vers la droite. Ex. : au → TM.
  - 2 lettres qui se trouvent dans la même colonne seront décalées d'une case vers le bas. Exemple : po → YQ.
  - Pour toutes les autres paires « l1 l2 » :
    - l1 sera remplacée par la lettre qui se trouve dans la ligne de l1 et la colonne de l2.
    - l2 sera remplacée par la lettre qui se trouve dans la ligne de l2 et la colonne de l1.
    - Ex. : ad → QB (coins d'un rectangle).

Système de Playfair (3)

Déchiffrement ?

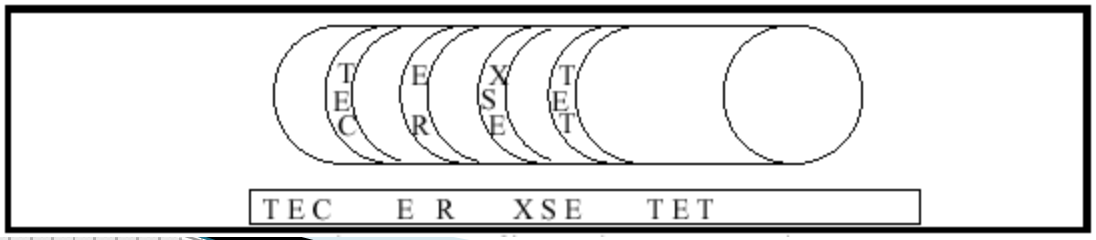
i	n	f	o	r
m	a	t	q	u
e	b	c	d	g
h	k	l	p	s
v	w	x	y	z

Chiffrement de substitution à longueur de clé égale à celle du texte (clés jetables)

- Pour éviter les attaques statistiques, il faut utiliser une substitution qui rend le texte chiffré non analysable statistiquement.
- Exemple de solution :
  - Générer une clé qui est une suite binaire parfaitement aléatoire (Phénomène physique aléatoire ou bruit électro magnétique).
  - Pour chiffrer un message, faire le « ou exclusif » du message et de la clé.
  - Si chaque clé ne sert qu'une fois, le chiffrement est incassable.

Chiffrement par transposition

- Principe **:** On procède à un réarrangement de l'ensemble des caractères (une transposition) qui cache le sens initial. La technique est très peu résistante aux attaques statistiques.
- Généralement, deux visions du texte géométriquement différentes.
- Exemple :
  - On enroule une langue de papier sur un tambour.
  - On écrit horizontalement un texte sur la lamelle enroulée.
  - Quand la lamelle est déroulée les lettres sont incompréhensibles.
  - Clé de dé/chiffrement : diamètre du cylindre.



Exemple de transposition à base matricielle

- Le message en clair est écrit dans une matrice.
- La clé est la matrice.
- La technique de transposition de base consiste à lire la matrice en colonne.
- Exemple (5,6):

M	E	S	S	A	G
E		S	E	C	R
E	T		A		T
R	A	N	S	P	O
S	E	R			

- Le message chiffré est donc: MEERSE TAESS NRSEAS AC P GRTO

Chiffrement à transposition avec chiffrement à substitution simple

- On combine la transposition avec une substitution et on réarrange l'ordre des colonnes selon une permutation qui est ajoutée à la matrice pour former la clé.
- Exemple :
  - Ordre d'exploration des colonnes 1 6 4 3 2 5, le texte chiffré est: « MEERSGRTO SEAS SS NRE TAEAC P »
  - On peut générer et mémoriser simplement des permutations en prenant une clé sous forme d'un mot qui ne comporte pas deux fois la même lettre.
  - On numérote les colonnes dans l'ordre ou apparaissent les lettres du mot dans l'alphabet.

Chiffrement à transposition (exemple)

- Un message chiffré :
  - C = PDECEE AAVAVR SATUG RNLEXS PACLUS DRSTO ESOSPE NEBIRR
- Trouver le message en clair M avec :
  - Clé K = matrice 6 x 8
  - lecture des colonnes 2-1-8-4-3-7-5-6

Chiffrement à transposition (exemple)

- Réponse :

APPRENDS A DANSER AVEC L OBSTACLE SI TU VEUX PROGRESSER

L'algorithme *Data Encryption Standard* : DES

Historique

- Dès le début des années 1960, la technologie des circuits intégrés permet de travailler à des circuits combinatoires complexes permettant d'automatiser:
  - la méthode de substitution.
  - la méthode de transposition.

=> Idée d'appliquer ces techniques en cascade dans un produit de chiffrement.

- Proposition IBM (1975)
- Adoption définitive et normalisation du DES d'IBM (1978) par le NBS (« National Bureau of Standards »).
- Normalisation ANSI X3.92 connue sous le nom de DEA (« *Data Encryption Algorithm* »).

Principes Généraux du DES

- Choix possibles pour la sécurité :
  - Méthodes simples de chiffrement et des clés très longues.
  - DES: Produit de transpositions et substitutions nombreuses et compliquées pour une clé relativement courte.

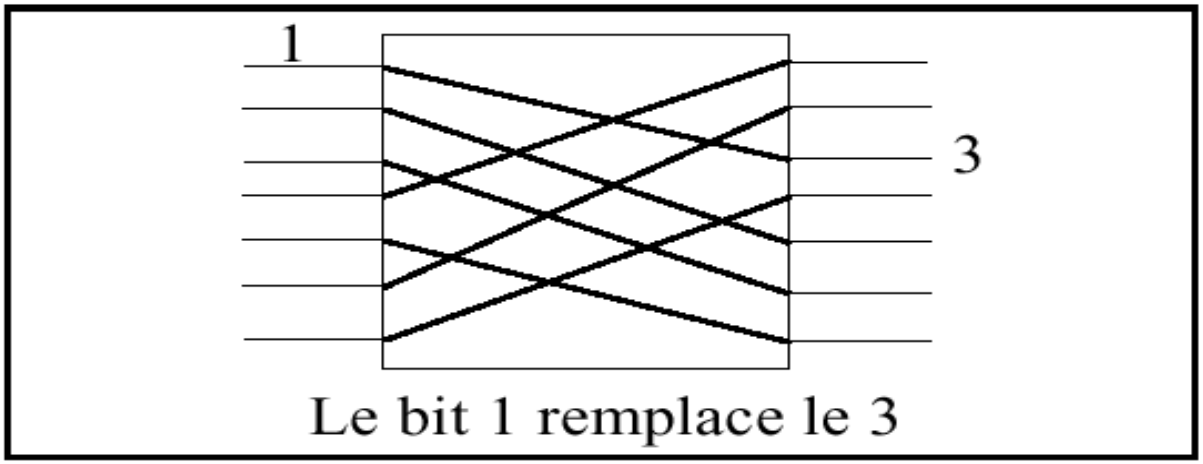
=> facilité de transport.

- Les chiffrements à substitution et à transposition sont faciles à réaliser en matériel.
  - Les boîtes de transposition « P-Box ».
  - Les boites de substitution « S-Box ».

Boîte de transposition
(P - box « Permutation box »)

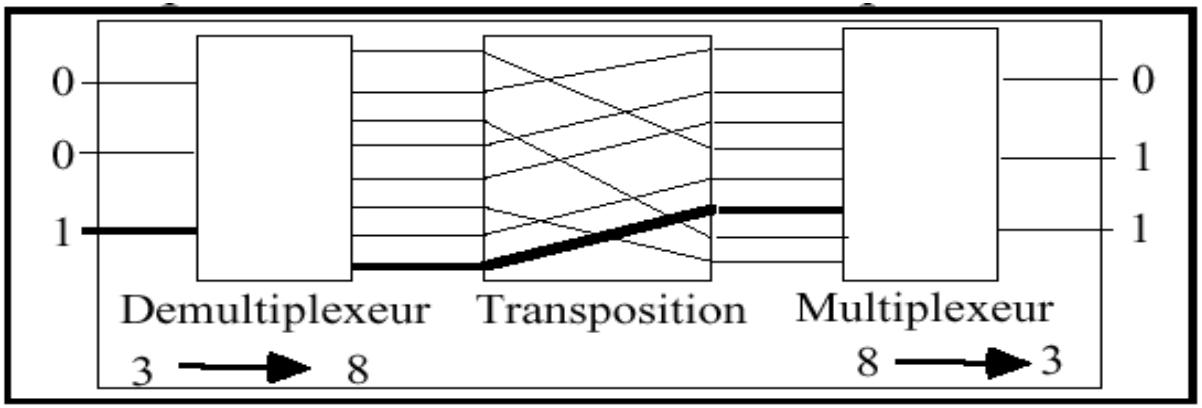
Facile à réaliser par simple câblage (matériel), ou par des tables (logiciel).

Exemple pour 8 bits (sol. matérielle).



Boîte de substitution (S - box)

- Exemple de solution matérielle pour 3 bits:
  - Trois bits sélectionnent un fil en sortie
  - L'ensemble subit une transposition.
  - Le résultat est remultiplexé sur 3 bits.

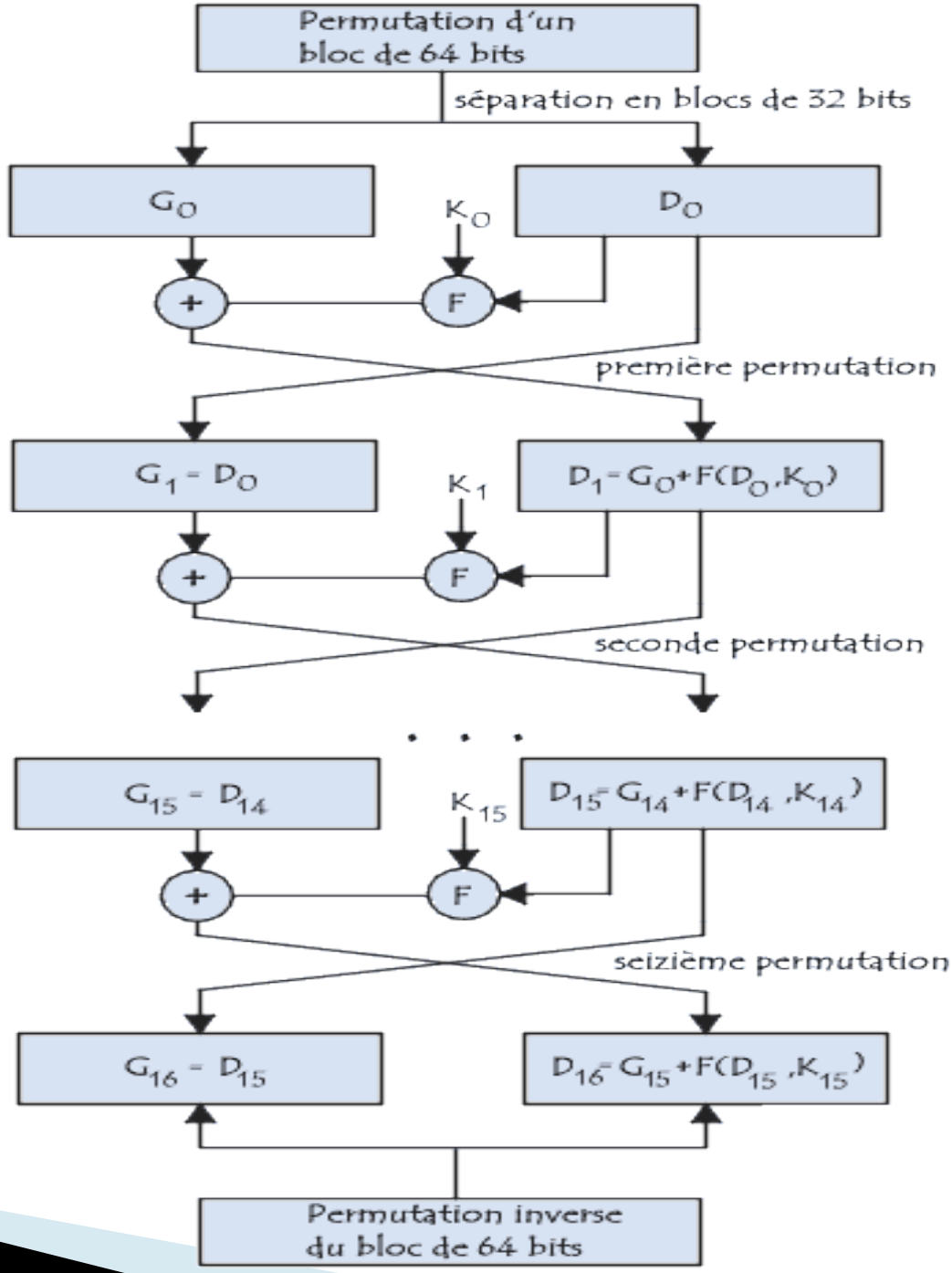


- Solution par consultation de table : Pour une configuration d'entrée on sélectionne directement au moyen d'une table la configuration de sortie.

Principes du DES

- Fractionnement du texte en blocs de 64 bits (8 octets) ;
- Clé de 64 bits de laquelle on enlève les 8 bits de parité.
  - Longueur effective de la clé: 56 bits.
- Permutation initiale des blocs.
- Découpage des blocs en deux parties: gauche et droite.
- Etapes de permutation et de substitution répétées 16 fois (appelées rondes/itérations).
- Recollement des parties gauche et droite puis permutation initiale inverse.

Schéma descriptif



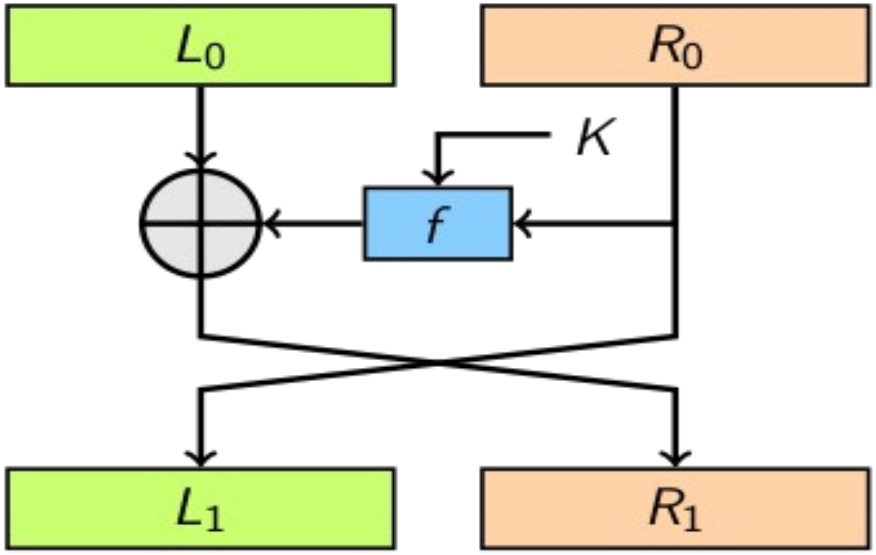
Fonctionnement du DES

- Le design du DES est basé sur deux concepts :
  - *Product ciphers* : chiffrement composé de plusieurs opérations simples. Ex.: transpositions, translations (e.g., XOR), transformations linéaires, opérations arithmétiques, multiplication, substitutions simples.

  - *Feistel ciphers* : transformation d'un texte clair de longueur 2t-bit (L_0 ;R_0), L_0 et R_0 de longueurs t-bit chacun, en un texte chiffré (L_r; R_r), via un processus de r-tours, où r>=1. Pour 1<=i<=r, tour i transforme (L_{i-1} ;R_{i-1}) en (L_i;R_i) en utilisant K_i comme suit : $L_i=R_{i-1}$, $R_i=L_{i-1}$ $f(R_{i-1};K_i)$, où K_i est dérivée à partir de la clé de chiffrement K.

Le schéma de Feistel

Cette transformation est bijective, car si on a un couple (L_1,R_1), on retrouve bien (L_0,R_0) par :
$R_0=L_1$ et $L_0=R_1$ XOR $f(L_1)$.



Fonctionnement du DES

Permutation initiale :

PI							
58	50	42	34	26	18	10	2
60	52	44	36	28	20	12	4
62	54	46	38	30	22	14	6
64	56	48	40	32	24	16	8
57	49	41	33	25	17	9	1
59	51	43	35	27	19	11	3
61	53	45	37	29	21	13	5
63	55	47	39	31	23	15	7

Fonctionnement du DES

Scindement en blocs de 32 bits :
G_0 contient tous les bits possédant une position paire dans le message initial, tandis que D_0 contient les bits de position impaire.

G_0							
58	50	42	34	26	18	10	2
60	52	44	36	28	20	12	4
62	54	46	38	30	22	14	6
64	56	48	40	32	24	16	8

D_0							
57	49	41	33	25	17	9	1
59	51	43	35	27	19	11	3
61	53	45	37	29	21	13	5
63	55	47	39	31	23	15	7

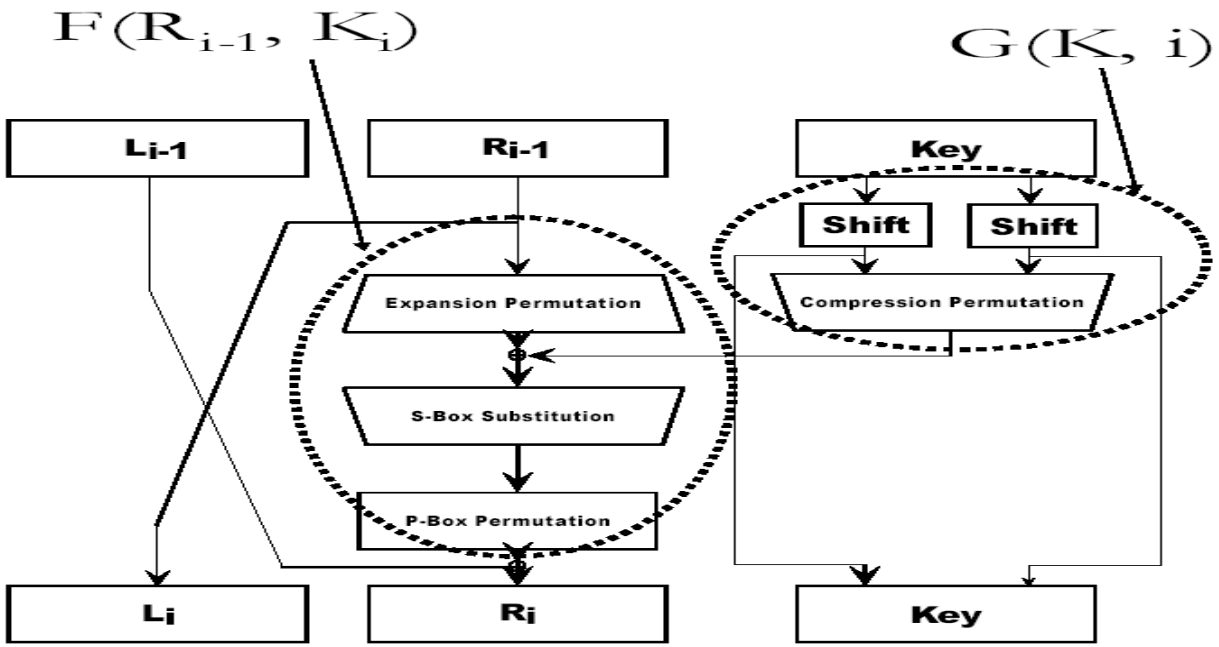
Fonctionnement du DES : itérations

$$i^{\text{ème}}\text{itération} \quad T_i = L_i R_i \begin{array}{l} \rightarrow L_i = t_1 \ldots t_{32} \\ \rightarrow R_i = t_{33} \ldots t_{64} \end{array}$$

$$\text{où} \quad L_i = R_{i-1}$$

$$R_i = L_{i-1} \oplus F(R_{i-1}, K_i)$$

$$K_i = G(K, i)$$



Fonction F : expansion

Les 32 bits du bloc D_0 sont étendus à 48 bits grâce à une table (matrice) appelé *table d'expansion*, dans laquelle les 48 bits sont mélangés et 16 d'entre eux sont dupliqués :

E					
32	1	2	3	4	5
4	5	6	7	8	9
8	9	10	11	12	13
12	13	14	15	16	17
16	17	18	19	20	21
20	21	22	23	24	25
24	25	26	27	28	29
28	29	30	31	32	1

Fonction F : addition avec la clé

- *OU exclusif* avec la première clé K_1 :

$$E(R_{i-1}) \oplus K_i = B_1 B_2 \quad \dots \quad B_8$$

B_j bloc de 6 bits

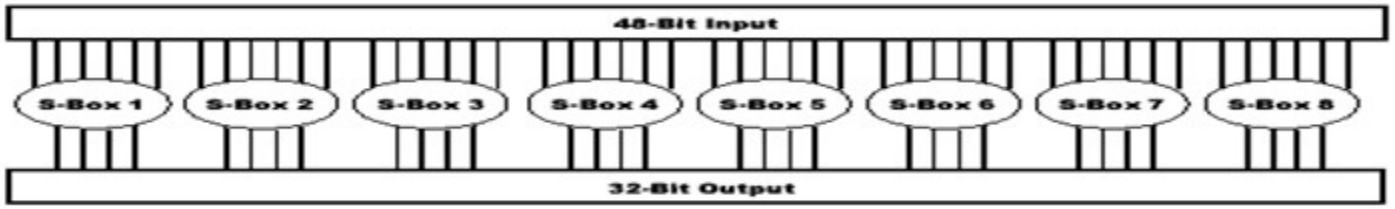
$B_j = b_1 b_2 b_3 b_4 b_5 b_6$

Fonction F : Fonction de substitution

- Le résultat est ensuite scindé en 8 blocs de 6 bits. Chaque bloc passe par des **fonctions de sélection** (appelées *boîtes de substitution*), notées $\mathbf{S_j}$.
- Les premiers et derniers bits de chaque bloc déterminent (en binaire) la ligne de la fonction de sélection, les autres bits (2, 3, 4 et 5) déterminent la colonne.
- Grâce à cette information, la fonction "sélectionne" une valeur codée sur 4 bits.

$$B_j \rightarrow S_j(B_j)$$

Bloc de 6 bits — Bloc de 4 bits



Fonction F : S-Box 1

Première fonction de substitution, représentée par :

S_1															
	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**
0	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
1	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
2	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
3	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

Le premier et le dernier bits de chaque bloc déterminent (en binaire) la ligne, les autres bits (2, 3, 4 et 5) déterminent la colonne.
Exemple : 101110 devient 1011 (11 en décimal).

S-Box

Row	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
1	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
2	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
3	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13
0	15	1	2	14	6	11	3	4	9	7	2	13	2	0	5	10
1	3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5
2	0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15
3	13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9
0	10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8
1	13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1
2	13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	7
3	1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12
0	7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15
1	13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9
2	10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4
3	3	15	0	6	10	1	13	8	9	4	5	11	12	7	2	14
0	2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	9
1	14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	6
2	4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	14
3	11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3
0	12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	11
1	10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8
2	9	14	15	5	2	8	12	3	7	0	4	10	1	13	1	6
3	4	3	2	12	9	5	15	10	11	14	1	7	6	0	8	13
0	4	11	2	14	15	0	8	13	3	12	9	7	5	10	6	1
1	13	0	11	7	4	9	1	10	14	3	5	12	2	15	8	6
2	1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2
3	6	11	13	8	1	4	10	7	9	5	0	15	14	2	3	12
0	13	2	8	4	6	15	11	1	10	9	3	14	5	0	12	7
1	1	15	13	8	10	3	7	4	12	5	6	11	0	14	9	2
2	7	11	4	1	9	12	14	2	0	6	10	13	15	3	5	8
3	2	1	14	7	4	10	8	13	15	12	9	0	3	5	6	11

Fonction F : permutation

Le bloc de 32 bits obtenu est soumis à une permutation P :

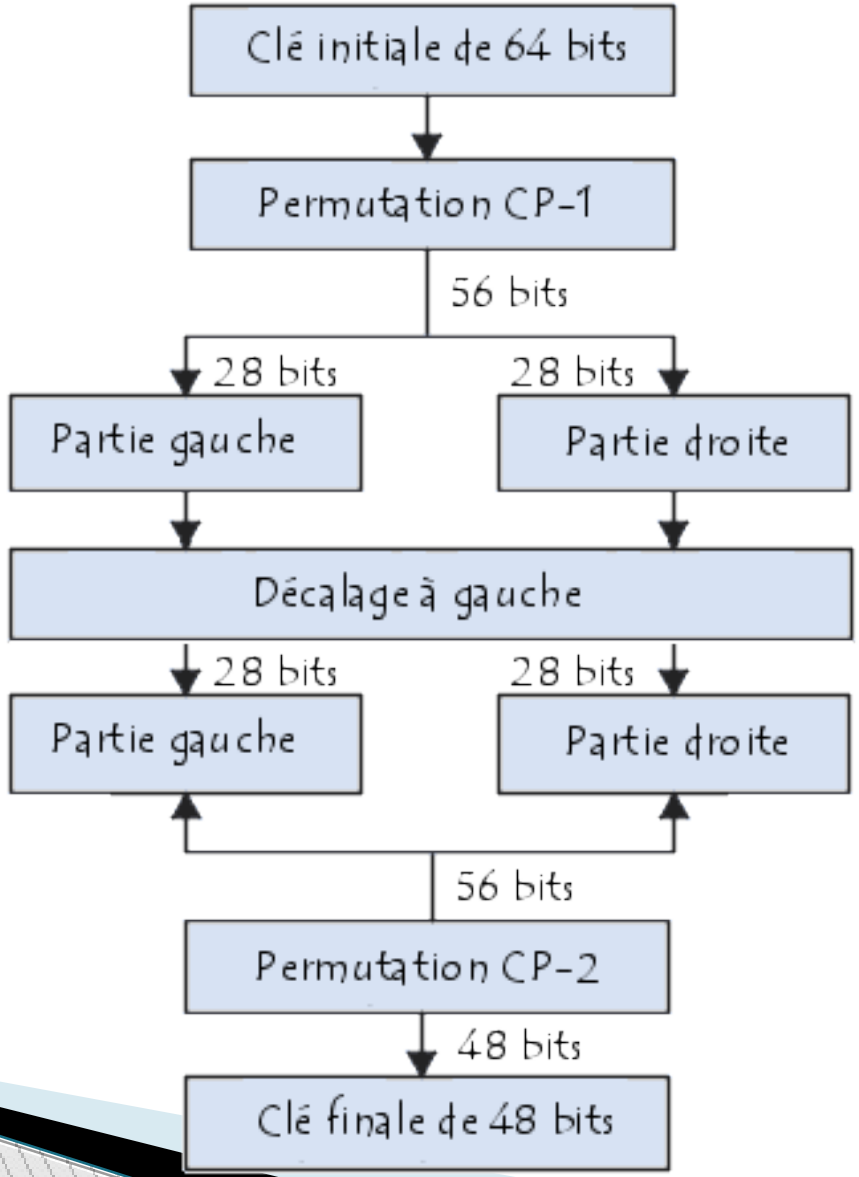
P							
16	7	20	21	29	12	28	17
1	15	23	26	5	18	31	10
2	8	24	14	32	27	3	9
19	13	30	6	22	11	4	25

Fonctionnement du DES : permutation initiale inverse

A la fin des itérations, les deux blocs $\mathbf{G}_{16}$ et $\mathbf{D}_{16}$ sont « recollés », puis soumis à la permutation initiale inverse :

PI-1							
40	8	48	16	56	24	64	32
39	7	47	15	55	23	63	31
38	6	46	14	54	22	62	30
37	5	45	13	53	21	61	29
36	4	44	12	52	20	60	28
35	3	43	11	51	19	59	27
34	2	42	10	50	18	58	26
33	1	41	9	49	17	57	25

Dérivation des Ki (fonction G)



Dérivation des Ki : permutation

- Les bits de parité de la clé sont éliminés afin d'obtenir une clé d'une longueur utile de 56 bits.
- La première étape consiste en une permutation notée **CP-1** :

CP-1													
57	49	41	33	25	17	9	1	58	50	42	34	26	18
10	2	59	51	43	35	27	19	11	3	60	52	44	36
63	55	47	39	31	23	15	7	62	54	46	38	30	22
14	6	61	53	45	37	29	21	13	5	28	20	12	4

Dérivation des Ki : découpage

La matrice peut en fait s'écrire sous la forme de deux matrice $\mathbf{G}_i$ et $\mathbf{D}_i$ composées chacune de 28 bits :

$\mathbf{G}_i$						
57	49	41	33	25	17	9
1	58	50	42	34	26	18
10	2	59	51	43	35	27
19	11	3	60	52	44	36

$\mathbf{D}_i$						
63	55	47	39	31	23	15
7	62	54	46	38	30	22
14	6	61	53	45	37	29
21	13	5	28	20	12	4

Dérivation des Ki : découpage

- Ces deux blocs subissent une rotation à gauche de 1 ou 2 position(s).
- Les 2 blocs de 28 bits sont ensuite regroupés en un bloc de 56 bits.
- Ce dernier passe par une permutation **CP-2**, fournissant en sortie un bloc de 48 bits, représentant la clé $\mathbf{K}_i$.

CP-2											
14	17	11	24	1	5	3	28	15	6	21	10
23	19	12	4	26	8	16	7	27	20	13	2
41	52	31	37	47	55	30	40	51	45	33	48
44	49	39	56	34	53	46	42	50	36	29	32

Dérivation des Ki : découpage

Des itérations de l'algorithme permettent de donner les 16 clés $\mathbf{K}_1$ à $\mathbf{K}_{16}$ utilisées dans l'algorithme du DES.

LS															
1	2	4	6	8	10	12	14	15	17	19	21	23	25	27	28

DES: Déchiffrement

- Même algorithme que pour le chiffrement mais utilisation des clés en ordre inverse K16 puis K15 puis … K1 à la 16ème itération:

$$R_{i-1} = L_i$$

$$L_{i-1} = R_i \oplus F(L_i, K_i)$$

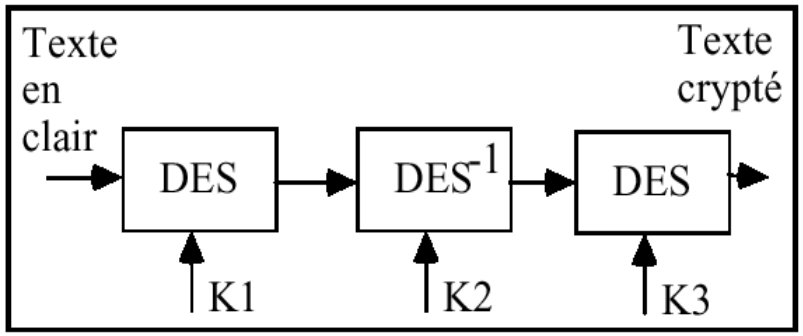
$$K_i = G(K, i)$$

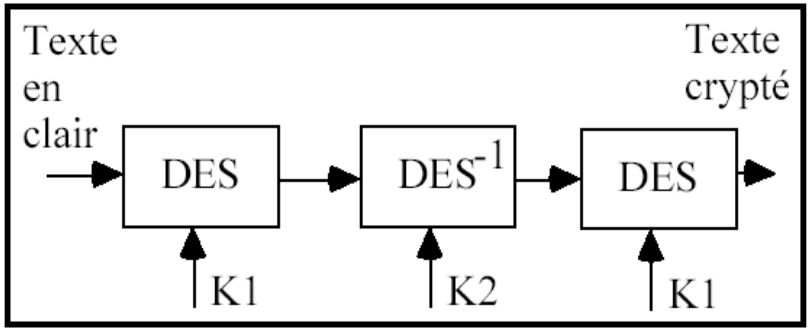
Les caractéristiques du DES

- Tous les bits de C dépendent de tous les bits de M.
- Effet d'avalanche: une légère modification de M ou de K entraîne une grande modification de C.
- Faiblesses du DES :
  - Les S-boxes pourraient contenir des failles mais les principes de leur choix n'ont jamais été rendus publiques.
  - La taille de la clé: des puces permettant l'essai de 10^6 clés/s ont été construites et peuvent être organisées en parallèle.
- Solution:
  - Augmenter la sécurité du DES sans réécrire l'algorithme.
  - Utiliser le DES 3 fois en série, avec 2 ou 3 clés différentes.

Le 3DES

- Permet d'augmenter la sécurité du DES, mais demande plus de ressources pour les chiffrement et le déchiffrement.
- Plusieurs types de chiffrement 3DES :
  - DES-EEE3 : 3 chiffrements DES avec 3 clés différentes.
  - DES-EDE3 : une clé différente pour chacune des 3 opérations DES (chiffrement, déchiffrement, chiffrement).
  - DES-EEE2 et DES-EDE2 : une clé différente pour la seconde opération (déchiffrement).





Attaques sur DES : Cryptanalyse différentielle

- Découverte par Eli Biham et Adi Shamir en 1991
- Permet de trouver la clé en utilisant 2^{47} textes clairs.
- Le principe est de disposer d'un DES implémenté dans une boîte noire hermétique avec une clé secrète à l'intérieur. En fournissant suffisamment de texte en entrée, on peut statistiquement analyser le comportement des sorties selon les entrées et retrouver la clé.
- Les entrées utilisées pour cette attaque doivent présenter une légère différence.

Attaques sur DES : cryptanalyse linéaire

- inventée par Mitsuru Matsui en 1993.
- Plus efficace mais moins pratique car l'attaquant ne dispose pas de la boîte noire et qu'il ne peut pas soumettre ses propres textes.
- Cette attaque nécessite 2^{43} couples (tous chiffrés avec la même clé) que l'attaquant a pu récupérer par un moyen ou un autre.
- Elle consiste à faire une approximation linéaire de DES en le simplifiant. En augmentant le nombre de couples disponibles, on améliore la précision de l'approximation et on peut en extraire la clé.

Attaques sur DES : compromis temps-mémoire

- Inventée par Martin Hellman au début des années 1980.
- En partant du principe que le même message va être chiffré plusieurs fois avec des clés différentes, on pourrait calculer une immense table qui contient toutes les versions chiffrées de ce message.
- Lorsque l'on intercepte un message chiffré, on peut le retrouver dans la table et obtenir la clé qui avait été utilisée pour le chiffrer.
- Cette attaque n'est bien sûr pas faisable car nous avons besoin d'une table de l'ordre du milliard de GB.
- Hellman a pu trouver un moyen pour réduire cette table à environ 1 téraoctet (soit 1 million de fois moins que la table complète), ce qui est faisable de nos jours.

Conclusion sur DES

- Standard assez ancien ayant bien tenu jusqu'à présent.
- Excellentes performances en vitesse de chiffrement.
- Niveau de sécurité pour une solution à clés privées très correct pour des applications ne nécessitant pas une confidentialité de haut niveau (militaire).

Travaux demandés

Chiffrement de substitution à longueur de clé égale à celle du texte (clés jetables) est système parfait : Présenter la démonstration de Shannon (2per)

Les attaques du DES (4 per)

Comparer la sécurité du DES et du double DES (2per)

L'algorithme *Advanced Encryption Standard* : AES

Cahier de charge

- En 1998, NIST (*National Institute of Standards and Technology*) lance un appel au public pour proposer un nouveau standard, satisfaisant les conditions:
  - Cryptosystème très robuste, à blocs et à clés symétriques pour utilisations gouvernementales et commerciales.
  - Plus efficace que le Triple DES
  - Plus sécurisant que le Triple DES
    - Taille des clés : 128, 192, et 256 bits
    - Taille des blocs: 128 bits (autres tailles optionnelles)
- Elaboré et évalué publiquement.
- Propriété intellectuelle libre dans le monde entier.

Cahier de charge

- Août 1999, 5 finalistes :
  - MARS *(IB*M): Complexe, rapide, haute marge de sécurité.
  - RC6 *(RSA Laboratorie*s): Très simple, très rapide, faible marge de sécurité.
  - Rijndael *(Joan Daemen*, *Vincent Rijme*n): Propre, rapide, bonne sécurité.
  - Serpent *(Ross Anderson*, *Eli Biham*, *Lars Knudse*n): Propre, lent, très haute marge de sécurité.
  - Twofish *(Bruce Schneier, John Kelsey*, *Doug Whiting*, *David Wagner, Niels Ferguson*, *Chris Hal*l): Complexe, très rapide, haute marge de sécurité.

Le choix de NIST

- L'algorithme retenu est Rijndael (contraction des noms des deux inventeurs belges, Vincent **Rij**men et Joan **Da**emen).
- Un bon compromis entre sécurité, performance, efficacité, facilité d'implémentation et flexibilité.
- Rijndael travaille par blocs de 128 bits et il est symétrique.
- La taille de la clé est généralement de 128 bits avec les variantes 192 et 256 bits.

Préliminaire mathématique (1)

GF(2^8) muni par les LCI (loi de composition interne) :

(+) est un xor.

(.) multiplication des polynômes modulo un polynôme irréductible $m(x)=x^8 + x^4 + x^3 + x + 1$.

Addition :

$$(x^6 + x^4 + x^2 + x + 1) + (x^7 + x + 1) = x^7 + x^6 + x^4 + x^2$$

$$\{01010111\} \oplus \{10000011\} = \{11010100\}$$

$$\{57\} \oplus \{83\} = \{d4\}$$

Préliminaire mathématique (2)

Multiplication

$$(x^6 + x^4 + x^2 + x + 1)\,(x^7 + x + 1) = x^{13} + x^{11} + x^9 + x^8 + x^7 + x^7 + x^5 + x^3 + x^2 + x + x^6 + x^4 + x^2 + x + 1$$

$$= x^{13} + x^{11} + x^9 + x^8 + x^6 + x^5 + x^4 + x^3 + 1$$

$$x^{13} + x^{11} + x^9 + x^8 + x^6 + x^5 + x^4 + x^3 + 1 \text{ modulo } (x^8 + x^4 + x^3 + x + 1)$$

$$= x^7 + x^6 + 1.$$

Comment ?

$$\{57\} \bullet \{83\} = \{c1\}$$

Préliminaire mathématique (3)

0000	0
0001	1
0010	2
0011	3

0100	4
0101	5
0110	6
0111	7

1000	8
1001	9
1010	a
1011	b

1100	c
1101	d
1110	e
1111	f

$$m(x) = x^8 + x^4 + x^3 + x + 1,$$

Préliminaire mathématique (4)

Identité de Bézout : Soit P et Q deux polynômes,
P et Q sont premiers entre eux si et seulement s'il existe deux polynômes V et U tels que :

$$PV + QU = 1.$$

Ainsi, tous les polynômes modulo m(x) sont inversibles.

Il y a une bijection entre les polynômes de degré inférieur à 8 et les éléments du $GF(2^8)$

$$P(x) . (R(x) + Q(x)) = P(x) . R(x) + P(x) . Q(x)$$

Préliminaire mathématique (5)

La multiplication par x cad par 02 :
P(x) . x = ?

Deux cas :

- Degrés de P(x) < 7 ainsi décalage à gauche (premier bit reçoit 0)
- Sinon le même décalage puis + 1b

Exercice
calculer : 57 . 13
Choisir la bonne réponse : d4.02 + bf.03 + 5d.01 + 30.01 =
1a , 05 , 04 , 12

Préliminaire mathématique (6)

xtime() : multiplication par x

$$\{57\} \bullet \{13\} = \ ?$$

$$\{57\} \bullet \{02\} = \text{xtime}(\{57\}) = \{ae\}$$

$$\{57\} \bullet \{04\} = \text{xtime}(\{ae\}) = \{47\}$$

$$\{57\} \bullet \{08\} = \text{xtime}(\{47\}) = \{8e\}$$

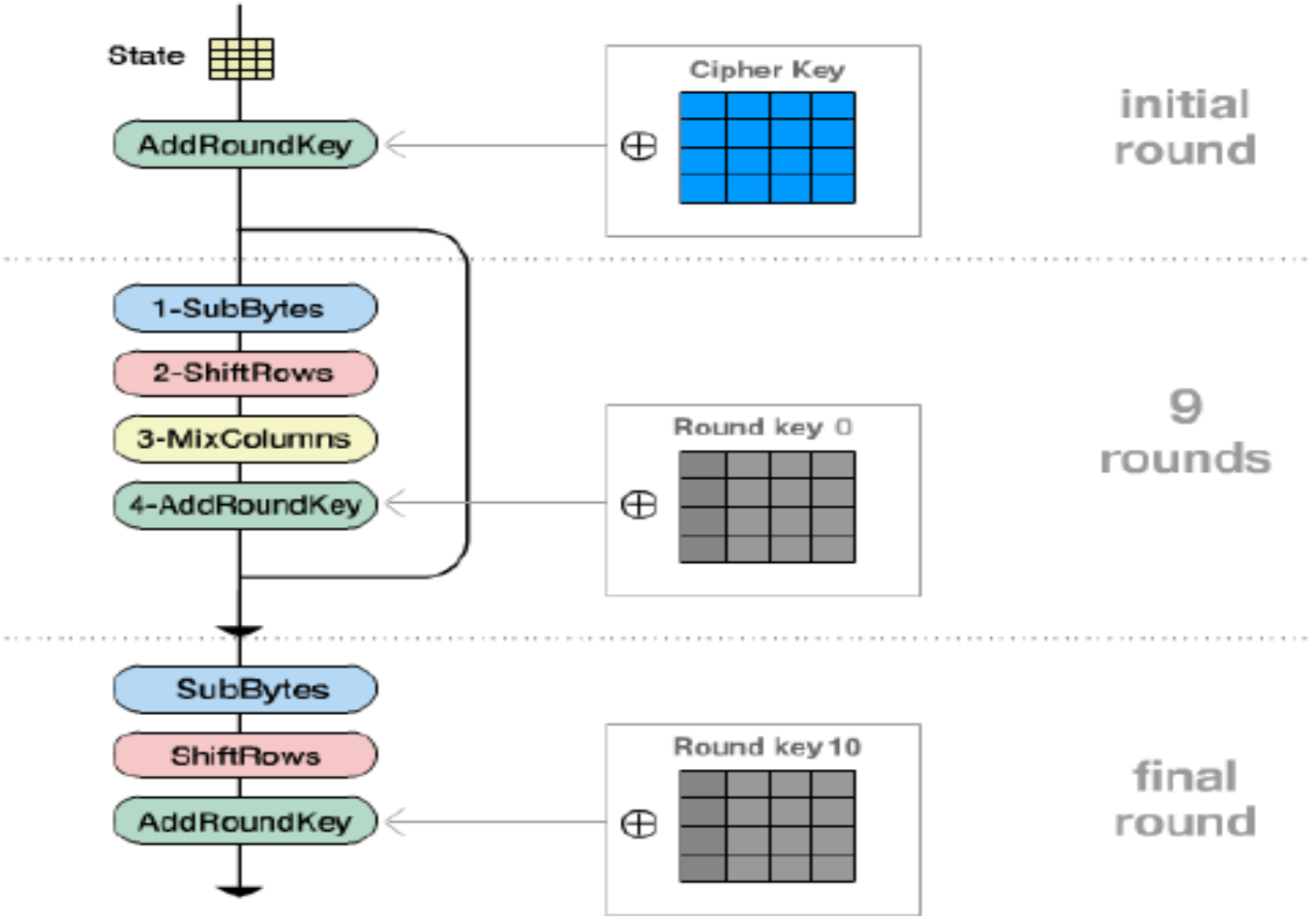
$$\{57\} \bullet \{10\} = \text{xtime}(\{8e\}) = \{07\},$$

$$\begin{aligned} \{57\} \bullet \{13\} &= \{57\} \bullet (\{01\} \oplus \{02\} \oplus \{10\}) \\ &= \{57\} \oplus \{ae\} \oplus \{07\} \\ &= \{fe\}. \end{aligned}$$

Structure de l'algorithme

- Algorithme itératif, pouvant être découpé en 3 blocs :
  - *Initial Roun*d: une seule opération : *Add Round Ke*y.
  - *N Round*s: N étant le nombre d'itérations. Ce nombre varie en fonction de la taille de la clé utilisée (128 bits N=9; 192 bits N=11; 256 bits N=13). Cette deuxième étape est constituée de N itérations comportant chacune les quatre opérations suivantes : *Sub Byte*s, *Shift Row*s, *Mix Column*s, *Add Round Ke*y.
  - *Final Roun*d. Cette étape est quasiment identique à l'une des N itérations de la deuxième étape. La seule différence est qu'elle ne comporte pas l'opération *Mix Column*s.

Schéma bloc de l'algorithme AES



Paramètres de l'algorithme

State	Intermediate Cipher result that can be pictured as a rectangular array of bytes, having four rows and ***Nb*** columns.
Word	A group of 32 bits that is treated either as a single entity or as an array of 4 bytes.
Nb	Number of columns (32-bit words) comprising the State. For this standard, ***Nb*** = 4.
Nk	Number of 32-bit words comprising the Cipher Key. For this standard, ***Nk*** = 4, 6, or 8.
Nr	Number of rounds, which is a function of ***Nk*** and ***Nb*** (which is fixed). For this standard, ***Nr*** = 10, 12, or 14.
Rcon[]	The round constant word array.

Déroulement du chiffrement

```
Cipher(byte in[4*Nb], byte out[4*Nb], word w[Nb*(Nr+1)])
begin
   byte  state[4,Nb]

   state = in

   AddRoundKey(state, w[0, Nb-1])

   for round = 1 step 1 to Nr-1
      SubBytes(state)
      ShiftRows(state)
      MixColumns(state)
      AddRoundKey(state, w[round*Nb, (round+1)*Nb-1])
   end for

   SubBytes(state)
   ShiftRows(state)
   AddRoundKey(state, w[Nr*Nb, (Nr+1)*Nb-1])

   out = state
end
```

Mise en forme des entrées

- Texte clair à chiffrer *(plaintex*t)
- Clé de chiffrement *(ke*y)
- Exemple : version 128bits de l'algorithme

Texte clair :
(*State*)

32	88	31	E0
43	5A	31	37
F6	30	98	07
A8	8D	A2	34

Clé :
(*Key*)

2B	28	AB	09
7E	AE	F7	CF
15	D2	15	4F
16	A6	88	3C

Première étape – *Round 0*

- Consiste à combiner la matrice *State* avec la clé: opération *Add Round Key*.
- Cette opération consiste à additionner modulo 2 (XOR) chaque octet de la matrice *State* avec son homologue de la matrice *Key*.
- On obtient ainsi la nouvelle matrice *State* (désigne l'état actuel du bloc en cours de chiffrement). Elle constitue la matrice d'entrée de l'étape suivante.

Deuxième étape – *Rounds 1-9*

- Substitution: Pour chaque élément de *state*,
  - le premier caractère hexadécimal indique une ligne de la *S-Box*.
  - le deuxième caractère indique une colonne.
- *Shift rwos*: Les décalages se font comme suit :
  - La première ligne n'est pas décalée.
  - La deuxième ligne est décalée de 1 octet vers la gauche.
  - La troisième ligne est décalée de 2 octets vers la gauche.
  - La quatrième ligne est décalée de 3 octets vers la gauche.
- *Mix Columns* **:** consiste à multiplier une matrice constante avec la matrice *State*.
- *Add Round Key* **:** une matrice *Key* est utilisée *(RoundN Key*) à partir de la matrice *key* pour disparaître la symétrie.

SubBytes (1)

- 0x42 devient 0x2c,
- 0x43 devient 0x1a,
- S-Box :

		y															
		0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
x	0	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
	1	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
	2	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
	3	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
	4	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
	5	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
	6	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
	7	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
	8	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
	9	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
	a	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
	b	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
	c	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
	d	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
	e	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
	f	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

SubBytes (2)

- S-Box est une fonction fixe et bijective de 8 bits vers 8 bits
- Définie comme un tableau à $2^8 = 256$ entrées
- Basée sur une opération algébrique qui s'écrit sous forme: S(X) = Affine(Inverse(X)) ou $S(X) = L \cdot 1/X + c$ où :

  - Inverse(X) est l'inverse multiplicatif de X dans GF(2^8),
  - L et C sont des constantes qui évitent les points fixes et particuliers.
  - + : Addition dans GF(2) :

$$b_i' = b_i \oplus b_{(i+4) \bmod 8} \oplus b_{(i+5) \bmod 8} \oplus b_{(i+6) \bmod 8} \oplus b_{(i+7) \bmod 8} \oplus c_i$$

  - c = {63} or {01100011}.

SubBytes (3)

Exemple

01 : son inverse est 01

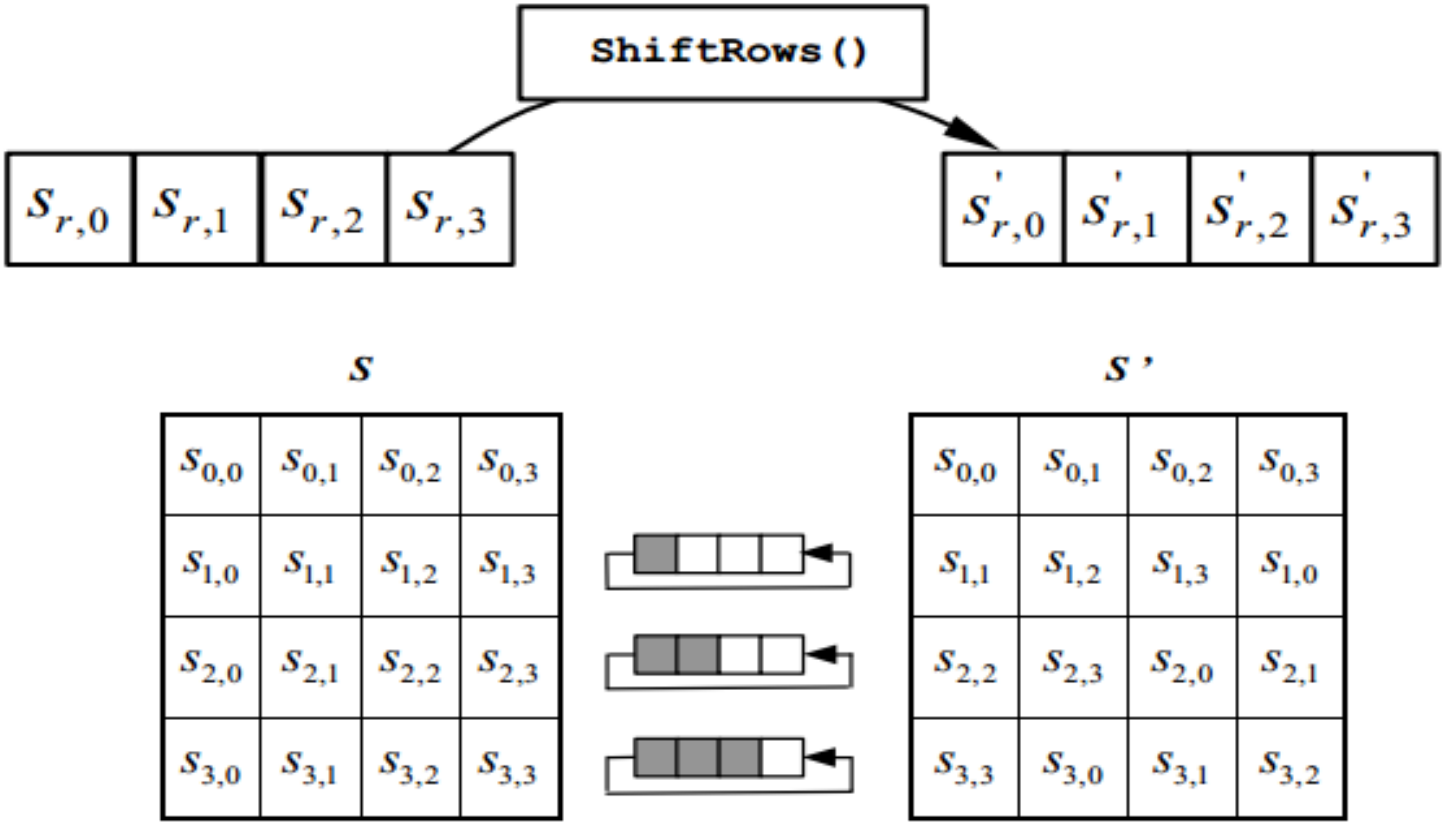
01 donne 0111 1100 : 7c

À refaire pour 02,

Puis f6 ?

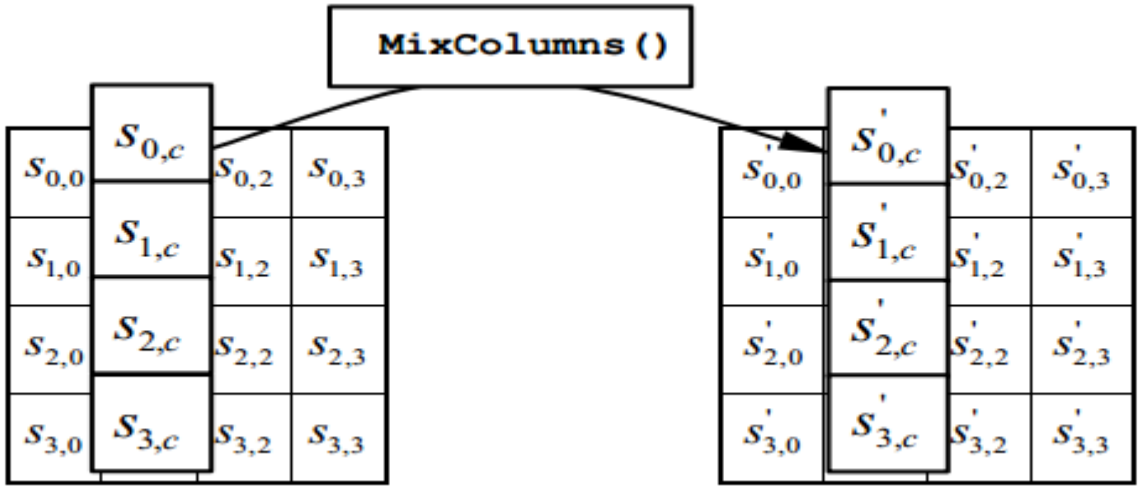
		$b_{i+4 \bmod 8}$	$b_{i+5 \bmod 8}$	$b_{i+6 \bmod 8}$	$b_{i+7 \bmod 8}$	63	S
b0	1	0	0	0	• 0	1	0
b1	0	0	0	0	1	1	0
b2	0	0	0	1	0	0	1
b3	0	0	1	0	0	0	1
b4	0	1	0	0	0	0	1
b5	0	0	0	0	0	1	1
b6	0	0	0	0	0	1	1
b7	0	0	0	0	0	0	0

ShiftRows


ShiftRows()
$s_{r,0}$ $s_{r,1}$ $s_{r,2}$ $s_{r,3}$
$s'_{r,0}$ $s'_{r,1}$ $s'_{r,2}$ $s'_{r,3}$
s
s'
$s_{0,0}$ $s_{0,1}$ $s_{0,2}$ $s_{0,3}$
$s_{1,0}$ $s_{1,1}$ $s_{1,2}$ $s_{1,3}$
$s_{2,0}$ $s_{2,1}$ $s_{2,2}$ $s_{2,3}$
$s_{3,0}$ $s_{3,1}$ $s_{3,2}$ $s_{3,3}$
$s_{0,0}$ $s_{0,1}$ $s_{0,2}$ $s_{0,3}$
$s_{1,1}$ $s_{1,2}$ $s_{1,3}$ $s_{1,0}$
$s_{2,2}$ $s_{2,3}$ $s_{2,0}$ $s_{2,1}$
$s_{3,3}$ $s_{3,0}$ $s_{3,1}$ $s_{3,2}$

MixColumns (1)

- Multiplier une matrice constante avec la matrice *State*.



$$\begin{bmatrix} s'_{0,c} \\ s'_{1,c} \\ s'_{2,c} \\ s'_{3,c} \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} s_{0,c} \\ s_{1,c} \\ s_{2,c} \\ s_{3,c} \end{bmatrix}$$

MixColumns (2)

- Multiplication modulo $X^4 + 1$ avec :
- $03\ x^3 + 01\ x^2 + 01\ x + 02$
- Pour tout polynôme a et b :

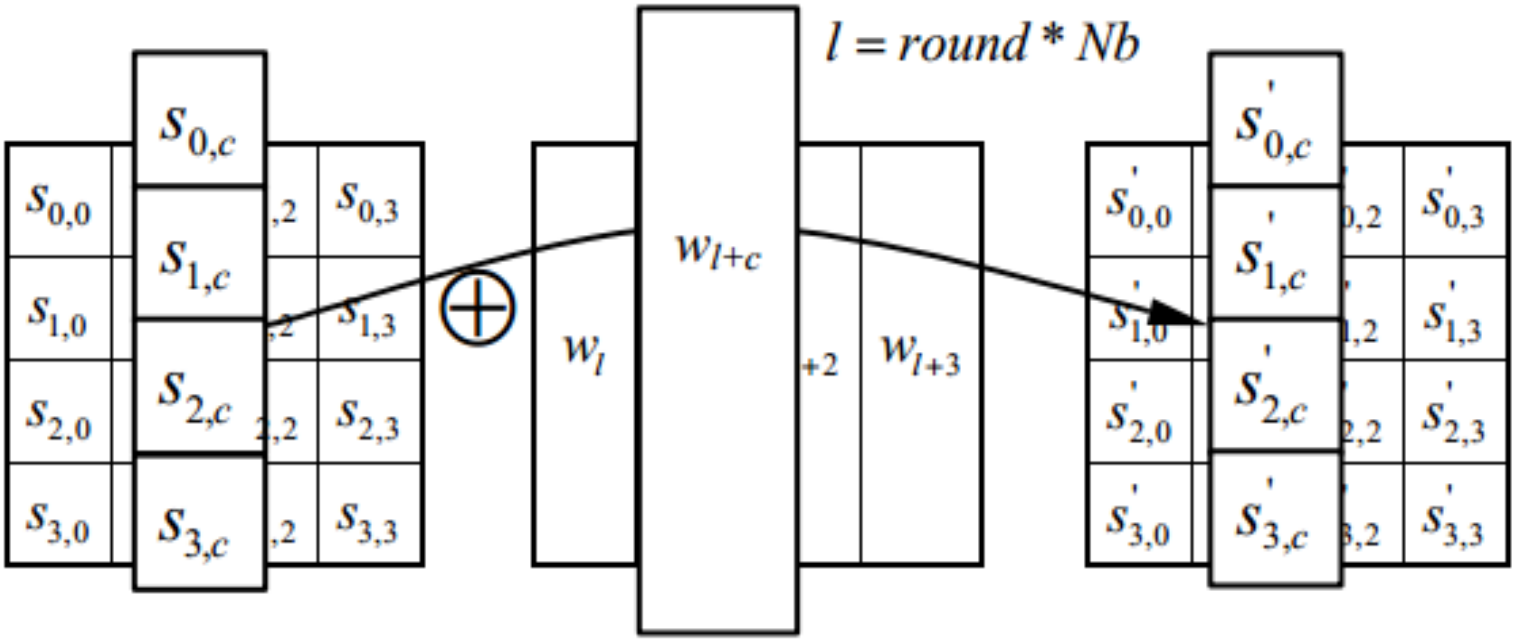
$$\begin{bmatrix} s'_{0,c} \\ s'_{1,c} \\ s'_{2,c} \\ s'_{3,c} \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} s_{0,c} \\ s_{1,c} \\ s_{2,c} \\ s_{3,c} \end{bmatrix}$$

a(x) . b(x) = d(x), notons que $x^i \bmod (x^4+1) = x^{i \bmod 4}$

$$d_0 = (a_0 \bullet b_0) \oplus (a_3 \bullet b_1) \oplus (a_2 \bullet b_2) \oplus (a_1 \bullet b_3)$$
$$d_1 = (a_1 \bullet b_0) \oplus (a_0 \bullet b_1) \oplus (a_3 \bullet b_2) \oplus (a_2 \bullet b_3)$$
$$d_2 = (a_2 \bullet b_0) \oplus (a_1 \bullet b_1) \oplus (a_0 \bullet b_2) \oplus (a_3 \bullet b_3)$$
$$d_3 = (a_3 \bullet b_0) \oplus (a_2 \bullet b_1) \oplus (a_1 \bullet b_2) \oplus (a_0 \bullet b_3)$$

$$\begin{bmatrix} d_0 \\ d_1 \\ d_2 \\ d_3 \end{bmatrix} = \begin{bmatrix} a_0 & a_3 & a_2 & a_1 \\ a_1 & a_0 & a_3 & a_2 \\ a_2 & a_1 & a_0 & a_3 \\ a_3 & a_2 & a_1 & a_0 \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{bmatrix}$$

AddRoundKey



Troisième étape – *Round 10*

- Quasiment identique à l'un des neuf *Rounds* de la deuxième étape.
- La seule différence est que, l'opération *Mix Columns* n'est pas effectuée.

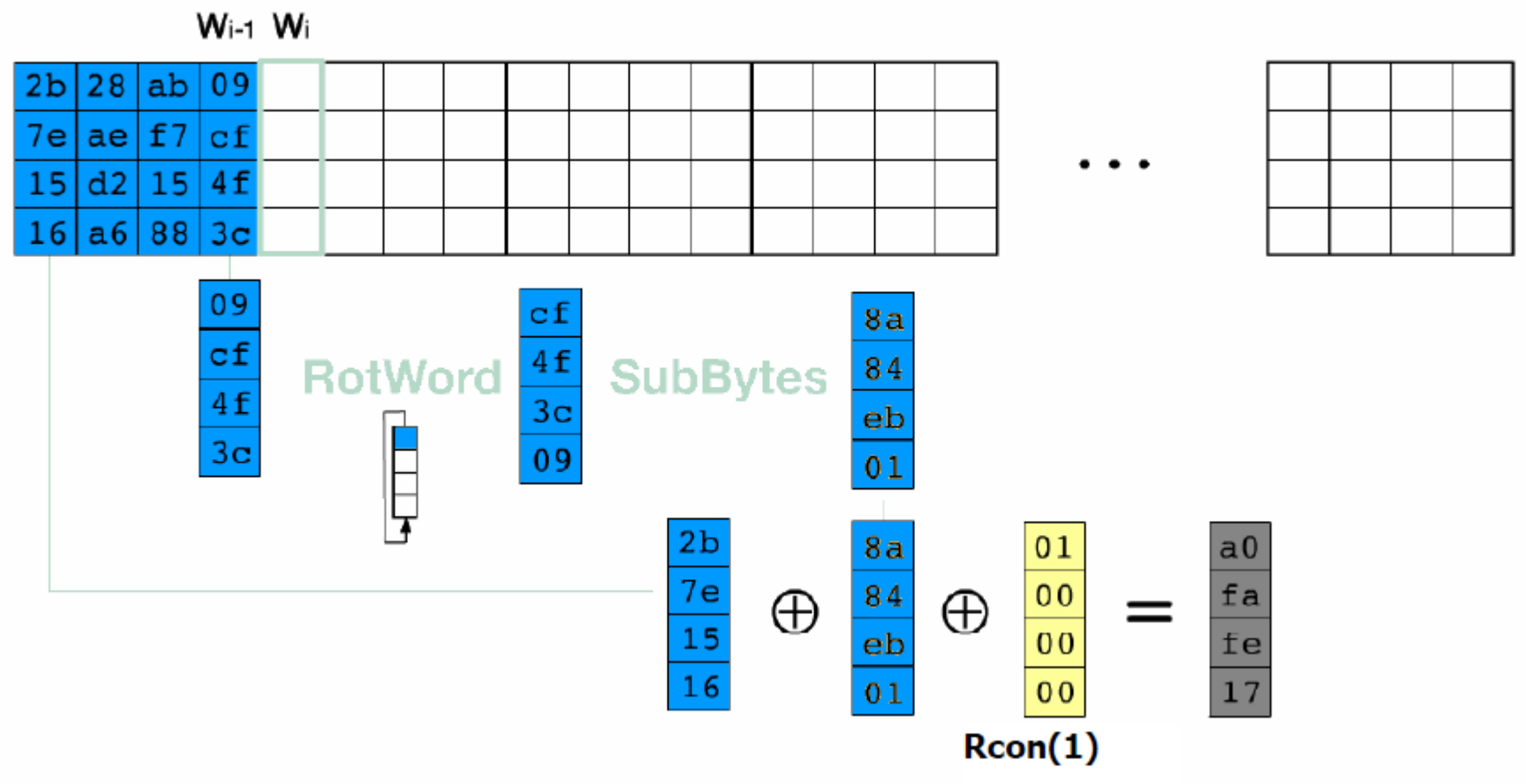
Diversification de la clé

```
KeyExpansion(byte key[4*Nk], word w[Nb*(Nr+1)], Nk)
begin
   word  temp

   i = 0

   while (i < Nk)
      w[i] = word(key[4*i], key[4*i+1], key[4*i+2], key[4*i+3])
      i = i+1
   end while

   i = Nk

   while (i < Nb * (Nr+1)]
      temp = w[i-1]
      if (i mod Nk = 0)
         temp = SubWord(RotWord(temp)) xor Rcon[i/Nk]
      else if (Nk > 6 and i mod Nk = 4)
         temp = SubWord(temp)
      end if
      w[i] = w[i-Nk] xor temp
      i = i + 1
   end while
end
```

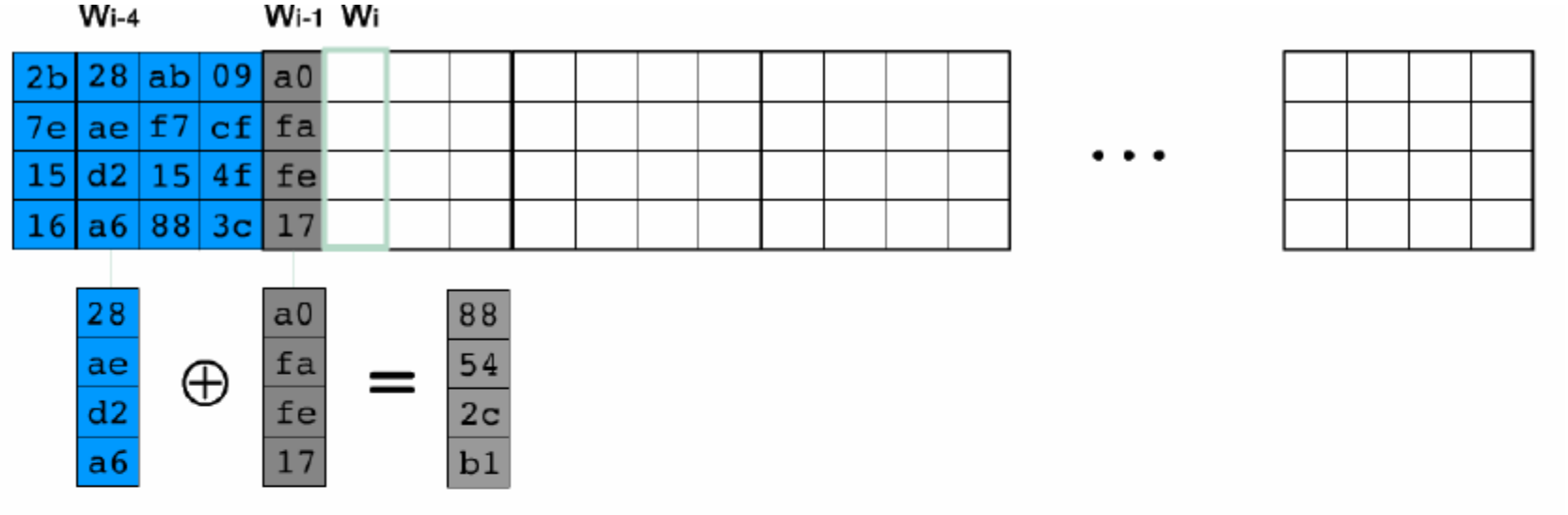
01	02	04	08	10	20	40	80	1b	36
00	00	00	00	00	00	00	00	00	00
00	00	00	00	00	00	00	00	00	00
00	00	00	00	00	00	00	00	00	00

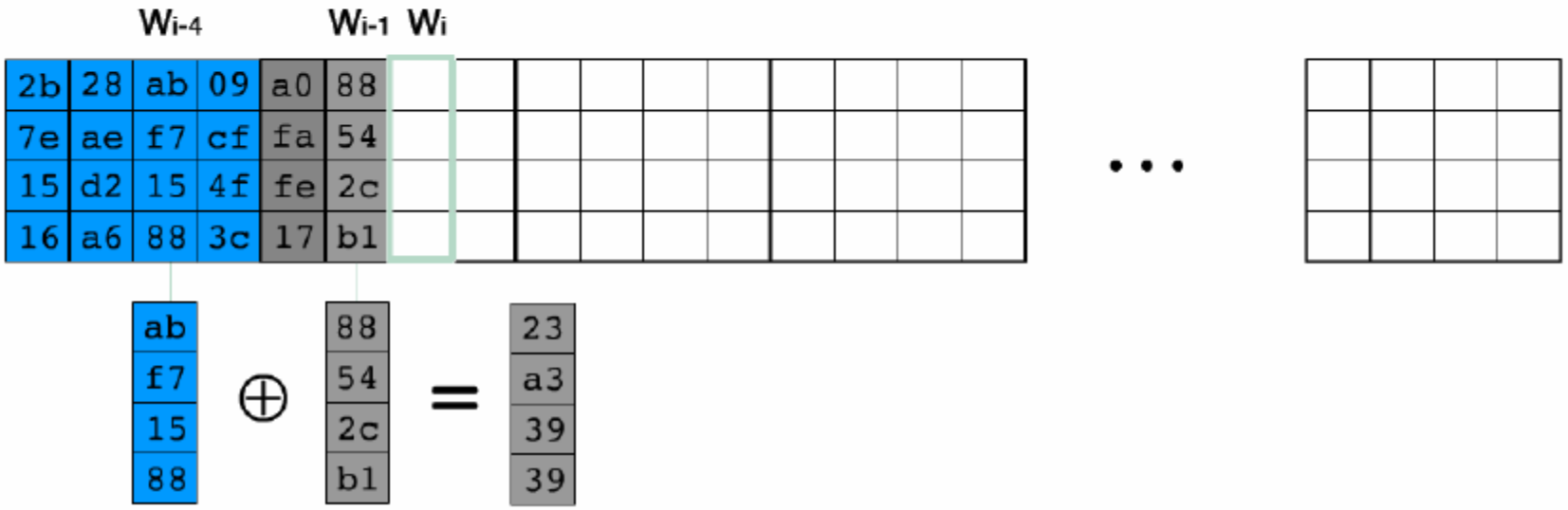
Rcon

Exemple (1)

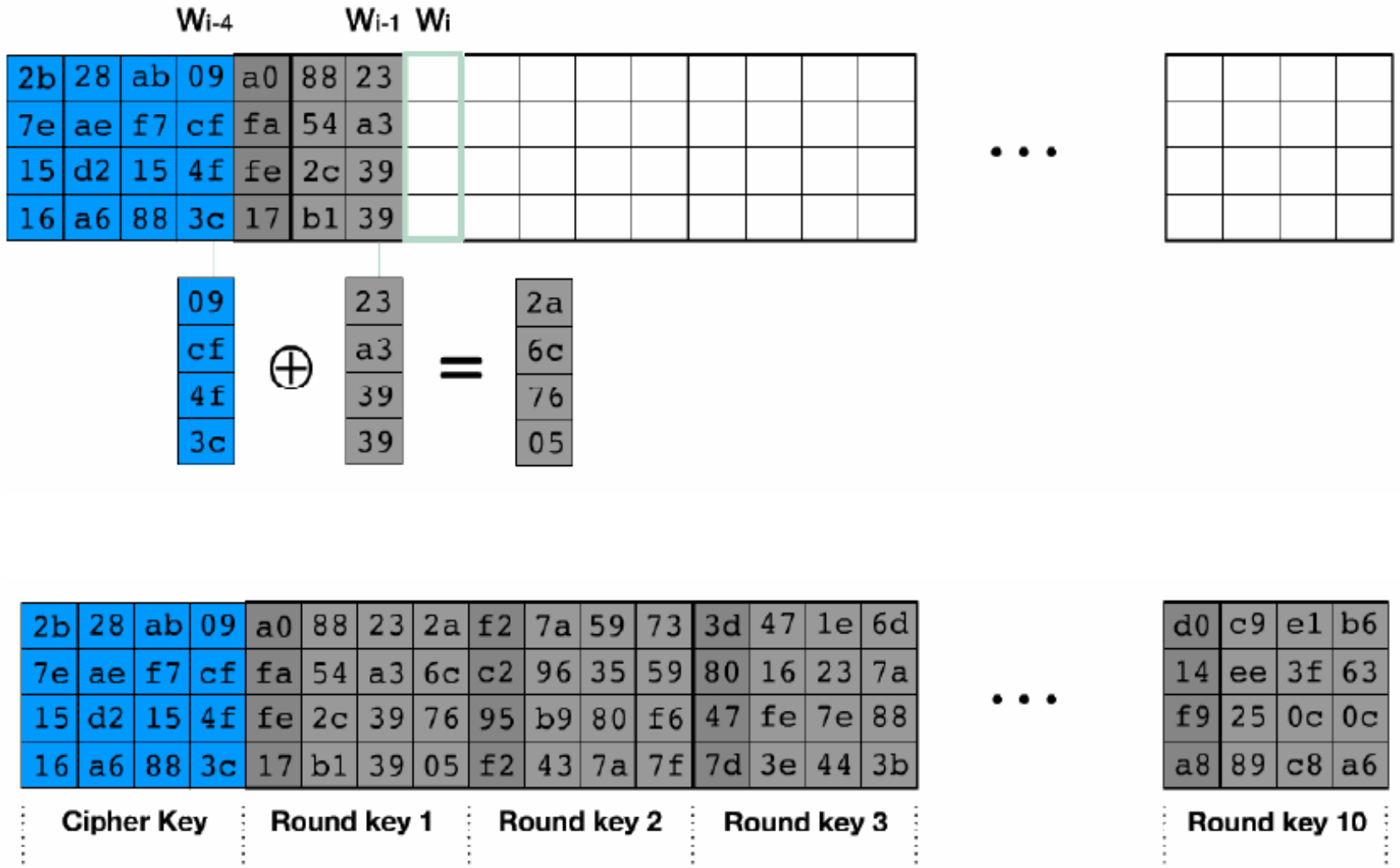

Wi-1 Wi
2b 28 ab 09
7e ae f7 cf
15 d2 15 4f
16 a6 88 3c
09 cf 4f 3c
RotWord
cf 4f 3c 09
SubBytes
8a 84 eb 01
2b 7e 15 16
8a 84 eb 01
01 00 00 00
Rcon(1)
a0 fa fe 17

Exemple (2)


Wi-4
Wi-1
Wi
2b 28 ab 09 a0
7e ae f7 cf fa
15 d2 15 4f fe
16 a6 88 3c 17
28 ae d2 a6
a0 fa fe 17
88 54 2c b1


Wi-4
Wi-1
Wi
2b 28 ab 09 a0 88
7e ae f7 cf fa 54
15 d2 15 4f fe 2c
16 a6 88 3c 17 b1
ab f7 15 88
88 54 2c b1
23 a3 39 39

Exemple (3)


Wi-4
Wi-1
Wi
2b 28 ab 09 a0 88 23
7e ae f7 cf fa 54 a3
15 d2 15 4f fe 2c 39
16 a6 88 3c 17 b1 39
09 cf 4f 3c
23 a3 39 39
2a 6c 76 05
2b 28 ab 09 a0 88 23 2a f2 7a 59 73 3d 47 1e 6d
7e ae f7 cf fa 54 a3 6c c2 96 35 59 80 16 23 7a
15 d2 15 4f fe 2c 39 76 95 b9 80 f6 47 fe 7e 88
16 a6 88 3c 17 b1 39 05 f2 43 7a 7f 7d 3e 44 3b
d0 c9 e1 b6
14 ee 3f 63
f9 25 0c 0c
a8 89 c8 a6
Cipher Key
Round key 1
Round key 2
Round key 3
Round key 10

Diversification de la clé (192 b)

AES : 128 ; 192 ; 256

AES 192 : w_0 w_1 w_2 w_3 w_4 w_5

12 rounds : besoin 48 word de la clé

Ainsi 48:6 =8 tours de diversification de clé
dernier Rcon = x^7 = 80

AES 256 ?

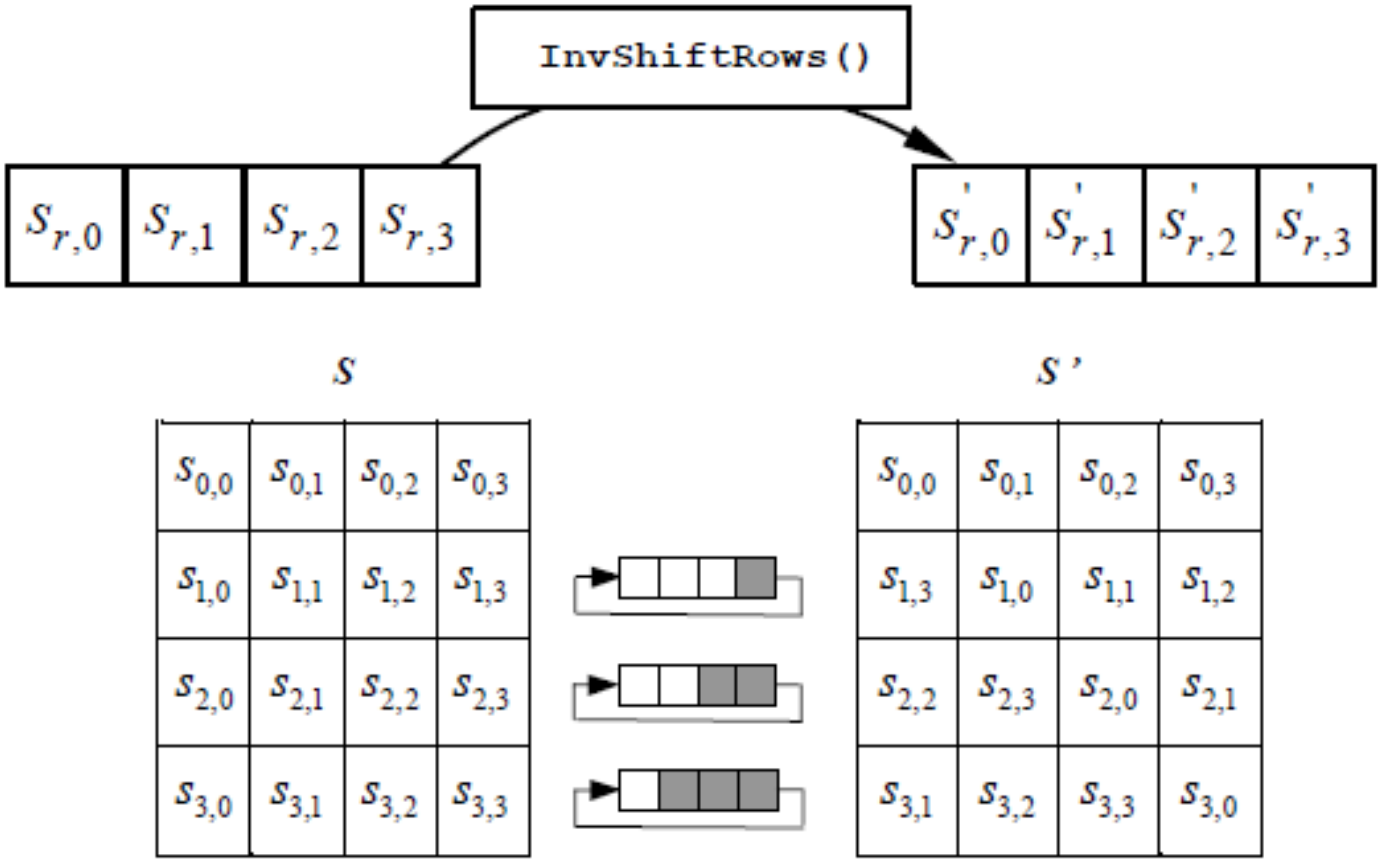
Diversification de la clé (256 b)

AES 256 : w_0 w_1 w_2 w_3 w_4 w_5 w_6 w_7

14 rounds : besoin 56 word de la clé

Ainsi 56:8 = 7 tours de diversification de clé
Le dernier Rcon = 40

Inv AES


InvShiftRows()
$S_{r,0}$ $S_{r,1}$ $S_{r,2}$ $S_{r,3}$
$S'_{r,0}$ $S'_{r,1}$ $S'_{r,2}$ $S'_{r,3}$
S
S'
$S_{0,0}$ $S_{0,1}$ $S_{0,2}$ $S_{0,3}$
$S_{1,0}$ $S_{1,1}$ $S_{1,2}$ $S_{1,3}$
$S_{2,0}$ $S_{2,1}$ $S_{2,2}$ $S_{2,3}$
$S_{3,0}$ $S_{3,1}$ $S_{3,2}$ $S_{3,3}$
$S_{0,0}$ $S_{0,1}$ $S_{0,2}$ $S_{0,3}$
$S_{1,3}$ $S_{1,0}$ $S_{1,1}$ $S_{1,2}$
$S_{2,2}$ $S_{2,3}$ $S_{2,0}$ $S_{2,1}$
$S_{3,1}$ $S_{3,2}$ $S_{3,3}$ $S_{3,0}$

InvSubBytes()

		y														
		0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
x	0	52	09	6a	d5	30	36	a5	38	bf	40	a3	9e	81	f3	d7	fb
	1	7c	e3	39	82	9b	2f	ff	87	34	8e	43	44	c4	de	e9	cb
	2	54	7b	94	32	a6	c2	23	3d	ee	4c	95	0b	42	fa	c3	4e
	3	08	2e	a1	66	28	d9	24	b2	76	5b	a2	49	6d	8b	d1	25
	4	72	f8	f6	64	86	68	98	16	d4	a4	5c	cc	5d	65	b6	92
	5	6c	70	48	50	fd	ed	b9	da	5e	15	46	57	a7	8d	9d	84
	6	90	d8	ab	00	8c	bc	d3	0a	f7	e4	58	05	b8	b3	45	06
	7	d0	2c	1e	8f	ca	3f	0f	02	c1	af	bd	03	01	13	8a	6b
	8	3a	91	11	41	4f	67	dc	ea	97	f2	cf	ce	f0	b4	e6	73
	9	96	ac	74	22	e7	ad	35	85	e2	f9	37	e8	1c	75	df	6e
	a	47	f1	1a	71	1d	29	c5	89	6f	b7	62	0e	aa	18	be	1b
	b	fc	56	3e	4b	c6	d2	79	20	9a	db	c0	fe	78	cd	5a	f4
	c	1f	dd	a8	33	88	07	c7	31	b1	12	10	59	27	80	ec	5f
	d	60	51	7f	a9	19	b5	4a	0d	2d	e5	7a	9f	93	c9	9c	ef
	e	a0	e0	3b	4d	ae	2a	f5	b0	c8	eb	bb	3c	83	53	99	61
	f	17	2b	04	7e	ba	77	d6	26	e1	69	14	63	55	21	0c	7d

InvMixColumns ()

$$a^{-1}(x) = \{0b\}x^3 + \{0d\}x^2 + \{09\}x + \{0e\}.$$

As described in Sec. 4.3, this can be written as a matrix multiplication. Let $s'(x) = a^{-1}(x) \otimes s(x)$:

$$\begin{bmatrix} s'_{0,c} \\ s'_{1,c} \\ s'_{2,c} \\ s'_{3,c} \end{bmatrix} = \begin{bmatrix} 0e & 0b & 0d & 09 \\ 09 & 0e & 0b & 0d \\ 0d & 09 & 0e & 0b \\ 0b & 0d & 09 & 0e \end{bmatrix} \begin{bmatrix} s_{0,c} \\ s_{1,c} \\ s_{2,c} \\ s_{3,c} \end{bmatrix} \quad \text{for } 0 \le c < Nb.$$

Conclusion sur AES

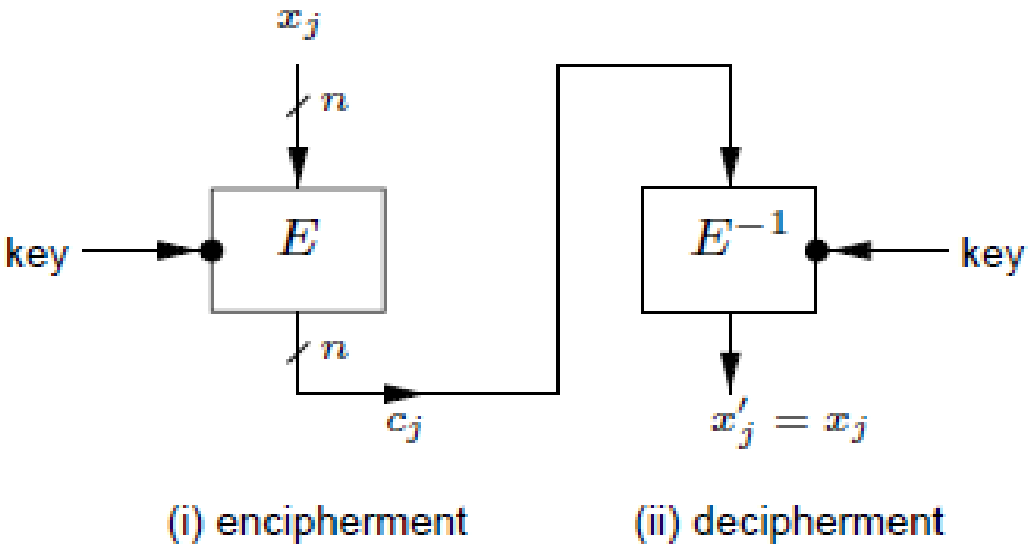
- L'utilisation de la *S-Box* constitue une réelle difficulté pour les cryptanalystes.
- L'opération *Mix Columns* combinée avec *Shift Rows* fait que, après les nombreux rounds, tous les bits de sortie dépendent de tous les bits d'entrée. Ceci aussi rend la cryptanalyse difficile.
- L'utilisation des clés secondaires construites par extension de la clé originale, quant à elle, complique les attaques liées à la clé en cassant les symétries.

Les modes de chiffrement

- Dans le cadre d'une implémentation pratique, l'algorithme 'pur' est combiné à une série d'opérations simples en vue d'améliorer la sécurité sans pour autant pénaliser l'efficacité de l'algorithme: « mode cryptographique **».**
- **Sécurité:**
  - Effacement des formats standards.
  - Protection contre la modification de C.
  - Chiffrement de plusieurs messages avec la même clé.
- **Efficacité:**
  - Le mode cryptographique ne pénalise pas l'efficacité du cryptosystème.
  - Limitation de la propagation des erreurs qui apparaissent dans M ou C.

Mode de chiffrement à livre de code électronique: *Electronic codebook mode*-ECB (1)

- Soient $X_1,X_2,\ldots,X_n$ blocs de textes clairs et k la clé :
  $C=C_1C_2\ldots C_n = E_k(X_1)E_k(X_2)\ldots E_k(X_n)$



Mode de chiffrement à livre de code électronique: *Electronic codebook mode*-ECB (2)

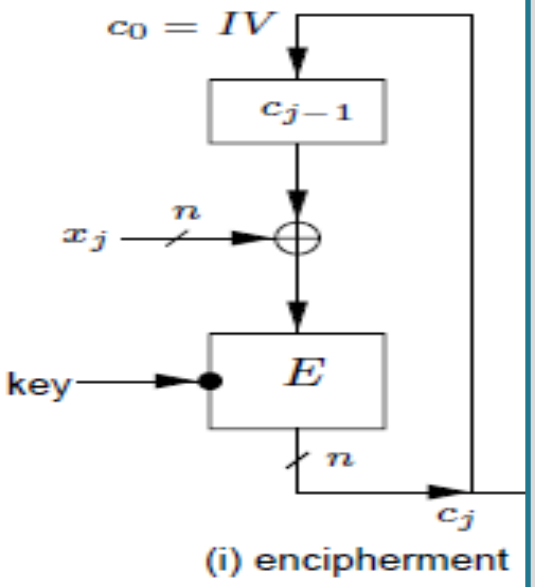
- Limitation de la propagation des erreurs:
  - Une erreur dans C_i n'affecte que X_i (le bloc entier).
  - La perte d'un bloc n'affecte pas les autres blocs.
- Sécurité:
  - Sécurité du cryptosystème repose entièrement sur le secret de la clé (Chaque bloc est chiffré indépendamment des autres).
  - En cas de redondance d'un même format de message (ex : email,…) l'attaquant les retrouve dans chaque message.
- Solution: enchaînement des blocs.

Vecteur d’initialization

- Définition : bloc de bits utilisé pour initialiser un état chiffrement.
- En général, une entrée aléatoire supplémentaire à l'algorithme de sorte que le résultat de chiffrement des mêmes données claires est différent pour un IV différent.
- Doit être connue par le destinataire. Différentes façons possibles :
  - Transmis avec le paquet de données;
  - Objet d'accord lors de l'échange de la clé;
  - Mesure de certains paramètres tels que la date courante;
  - Adresse de l'expéditeur ou du destinataire;
  - Numéro de dossier, industrie ou d'autres données, etc.
  - Ou encore : plusieurs variables reliées entre elles par un algorithme de hachage.

Mode de chiffrement par blocs enchaînés: *Cipher Block Chaining Mode*-CBC (1)

- Chaque bloc C_j dépend de X_j et de C_{j-1} : $C_j=E_k(X_j\ C_{j-1})$

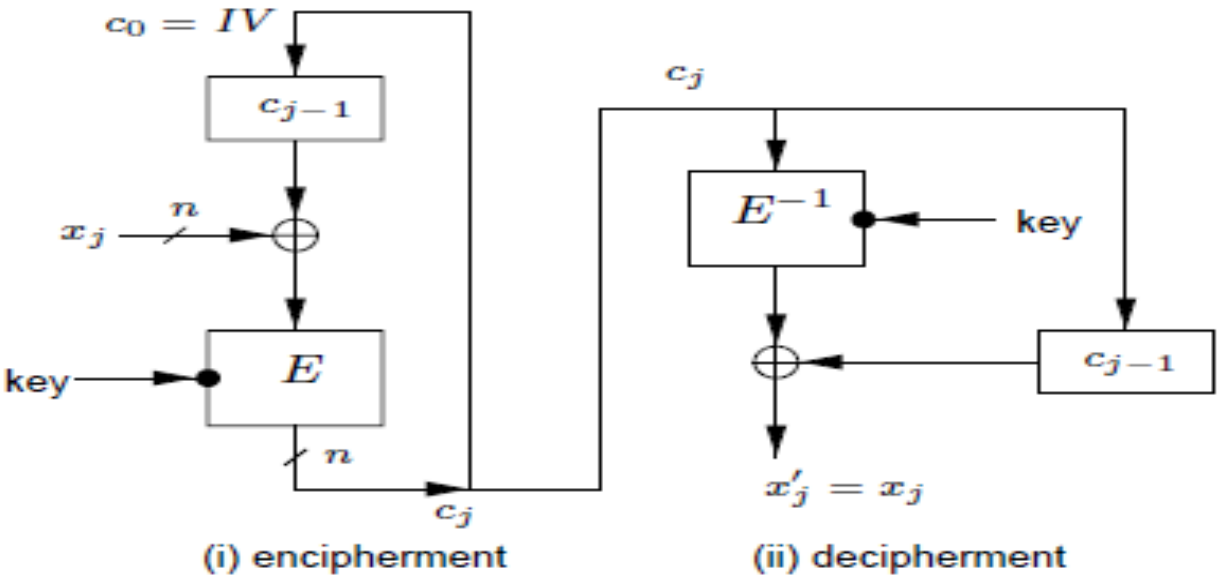


(i) encipherment

Déchiffrement ?

Mode de chiffrement par blocs enchaînés: *Cipher Block Chaining Mode*-CBC (2)

$C_j=E_k(X_j\ C_{j-1})$ et $X_j= C_{j-1}\ D_k(C_j)$



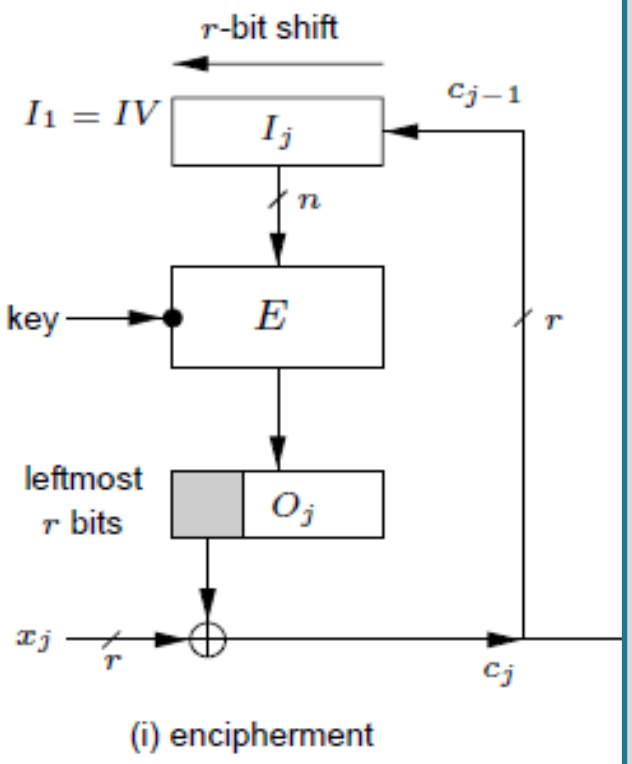
Mode de chiffrement par blocs enchaînés: *Cipher Block Chaining Mode*-CBC (3)

- Chaque message est précédé d'un Vecteur d'Initialisation (IV) unique, aléatoire.
- Sécurité:
  - Si X =X' alors C C' (avec même k et IV).
  - Effacement des formats standards grâce à l'enchaînement.
  - Il n'y a plus de risques de répétition de blocs.
- Propagation d'erreurs ?

Mode de chiffrement par blocs enchaînés: *Cipher Block Chaining Mode*-CBC (4)

- Propagation d'erreurs:
  - Une erreur dans X_i modifie tous les C_i suivants mais ne se retrouve qu'en X_i après déchiffrement.
  - Une erreur dans C_i lors de la communication affecte le bloc X_i entier plus un bit du bloc X_{i+1} .
  - La perte ou l'ajout d'un bit de C_i affecte tous les blocs suivants après déchiffrement (X_i, X_{i+1}…) => Perte des limites de blocs.

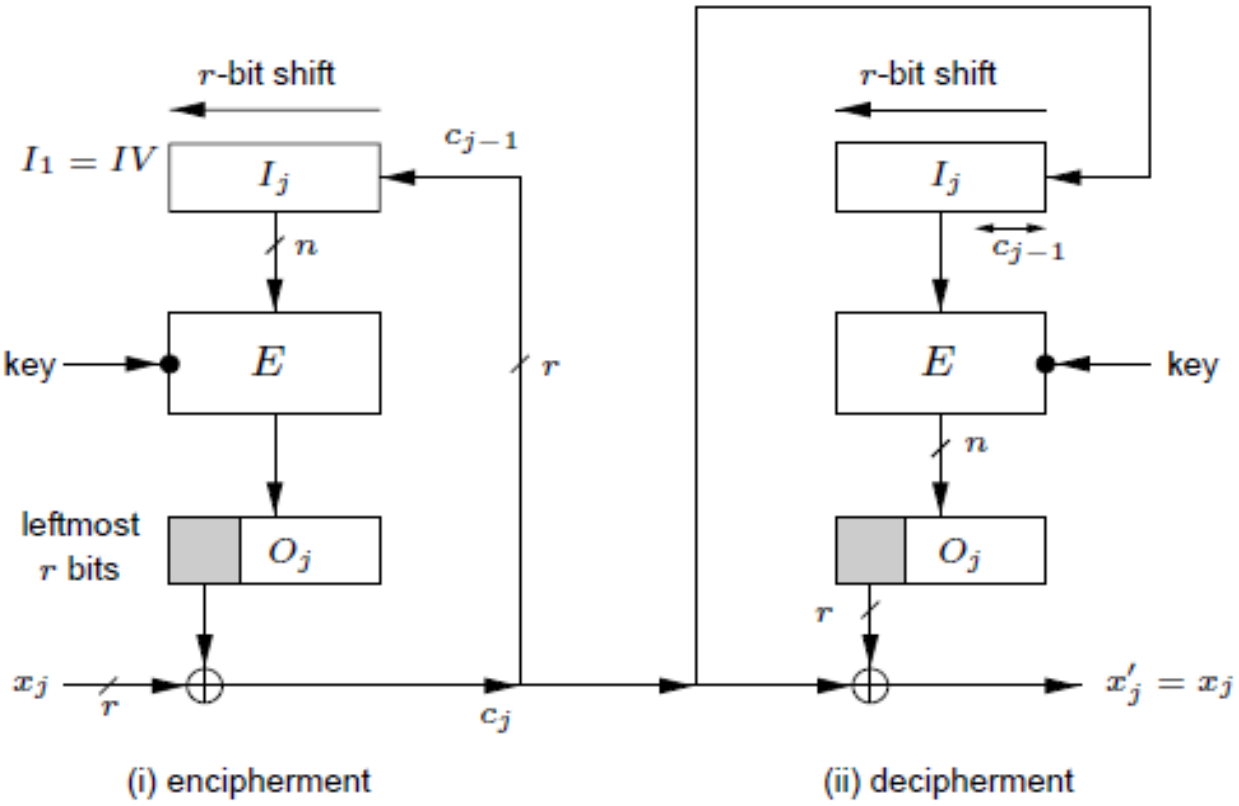
Mode de chiffrement par blocs avec contre réaction: *Cipher-Feedback Mode*-CFB (1)



(i) encipherment

Déchiffrement ?

Mode de chiffrement par blocs avec contre réaction: *Cipher-Feedback Mode*-CFB (2)



Mode de chiffrement par blocs avec contre réaction: *Cipher-Feedback Mode*-CFB (3)

Algorithm CFB mode of operation (CFB-r)

INPUT: k-bit key K; n-bit IV; r-bit plaintext blocks $x_1, \ldots, x_u$ $(1 \leq r \leq n)$.

SUMMARY: produce r-bit ciphertext blocks $c_1, \ldots, c_u$; decrypt to recover plaintext.

1. Encryption: $I_1 \leftarrow IV$. (I_j is the input value in a shift register.) For $1 \leq j \leq u$:
   (a) $O_j \leftarrow E_K(I_j)$. (Compute the block cipher output.)
   (b) $t_j \leftarrow$ the r leftmost bits of O_j. (Assume the leftmost is identified as bit 1.)
   (c) $c_j \leftarrow x_j \oplus t_j$. (Transmit the r-bit ciphertext block c_j.)
   (d) $I_{j+1} \leftarrow 2^r \cdot I_j + c_j \bmod 2^n$. (Shift c_j into right end of shift register.)
2. Decryption: $I_1 \leftarrow IV$. For $1 \leq j \leq u$, upon receiving c_j:
   $x_j \leftarrow c_j \oplus t_j$, where t_j, O_j and I_j are computed as above.

Mode de chiffrement par blocs avec contre réaction: *Cipher-Feedback Mode*-CFB (4)

- Le registre à décalage est initialisé avec un vecteur d'initialisation.
- Le bloc complet est alors chiffré.
- L'octet de poids fort du texte chiffré est combiné par un **ou exclusif** avec l'octet de texte en clair.
- Le résultat de cette opération est alors transmis en même temps qu'il est injecté dans le registre à décalage.

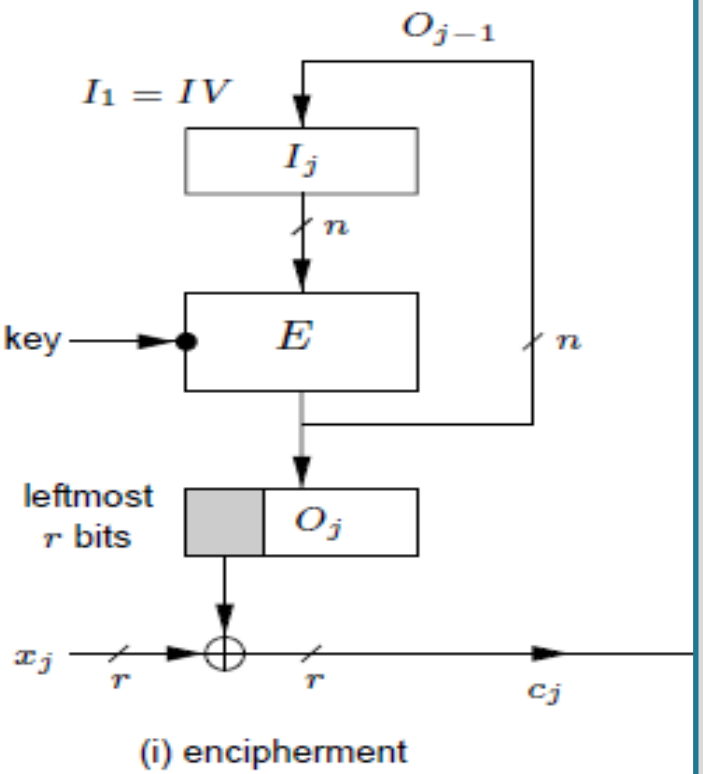
Mode de chiffrement par blocs avec contre réaction: *Cipher-Feedback Mode*-CFB (5)

- Efficacité:
  - Le chiffrement/déchiffrement peut débuter après réception de sous-blocs de plus petite taille (ex. 8b au lieu de 64b).
  - Intéressant pour communications chiffrées dans un réseau.
- Propagation d'erreur
  - Une erreur dans X_i affecte les 64/k C_i suivants mais ne se répercute que dans le X_i concerné après déchiffrement.
  - Une erreur dans C_i lors de la communication affecte le X_i correspondant après déchiffrement mais aussi les 64/k blocs suivants (tant que le C_i est présent dans le registre).

Mode de chiffrement par blocs avec contre réaction: *Cipher-Feedback Mode*-CFB (6)

- Perte d'un bloc C_i : Le synchronisme est récupéré dès que C_i est « sorti » du registre.
- Chaque message est précédé d'un IV unique, aléatoire.
- Sécurité:
  - Si $X_i = X_{i'}$ alors $C_i \quad C_{i'}$ (avec même k et IV).
  - Effacement des formats standards grâce à l'enchaînement.
  - Il ne reste plus de risques de répétition de blocs.

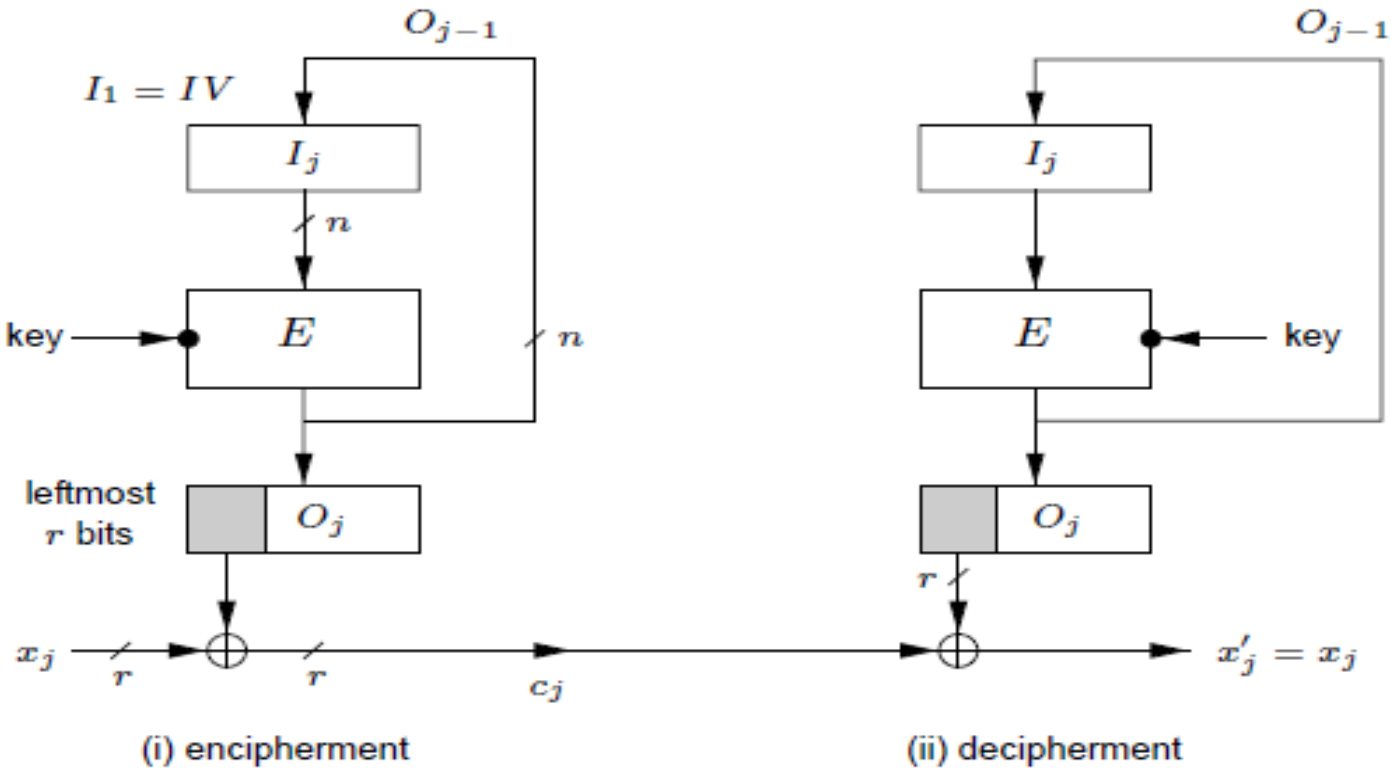
Mode de chiffrement avec contre-réaction du bloc de sortie:*Output-Feedback Mode*-OFB (1)



(i) encipherment

Déchiffrement ?

Mode de chiffrement avec contre-réaction du bloc de sortie:*Output-Feedback Mode*-OFB (2)



Mode de chiffrement avec contre-réaction du bloc de sortie:*Output-Feedback Mode*-OFB (3)

Algorithm OFB mode with full feedback (per ISO 10116)

INPUT: k-bit key K; n-bit IV; r-bit plaintext blocks $x_1, \ldots, x_u$ $(1 \leq r \leq n)$.
SUMMARY: produce r-bit ciphertext blocks $c_1, \ldots, c_u$; decrypt to recover plaintext.

1. Encryption: $I_1 \leftarrow IV$. For $1 \leq j \leq u$, given plaintext block x_j:
   (a) $O_j \leftarrow E_K(I_j)$. (Compute the block cipher output.)
   (b) $t_j \leftarrow$ the r leftmost bits of O_j. (Assume the leftmost is identified as bit 1.)
   (c) $c_j \leftarrow x_j \oplus t_j$. (Transmit the r-bit ciphertext block c_j.)
   (d) $I_{j+1} \leftarrow O_j$. (Update the block cipher input for the next block.)
2. Decryption: $I_1 \leftarrow IV$. For $1 \leq j \leq u$, upon receiving c_j:
   $x_j \leftarrow c_j \oplus t_j$, where t_j, O_j, and I_j are computed as above.

Mode de chiffrement avec contre-réaction du bloc de sortie:*Output-Feedback Mode*-OFB (4)

- Mode semblable à CFB excepté que les registres à contre-réaction contiennent les n-bits à gauche du résultat du chiffrement du DES.
  - Simplicité : mécanisme de contre-réaction est indépendant de X_i et C_i.
- Efficacité:
  - L'algorithme cryptographique peut être lancé off-line et avant même que le message M n'existe.

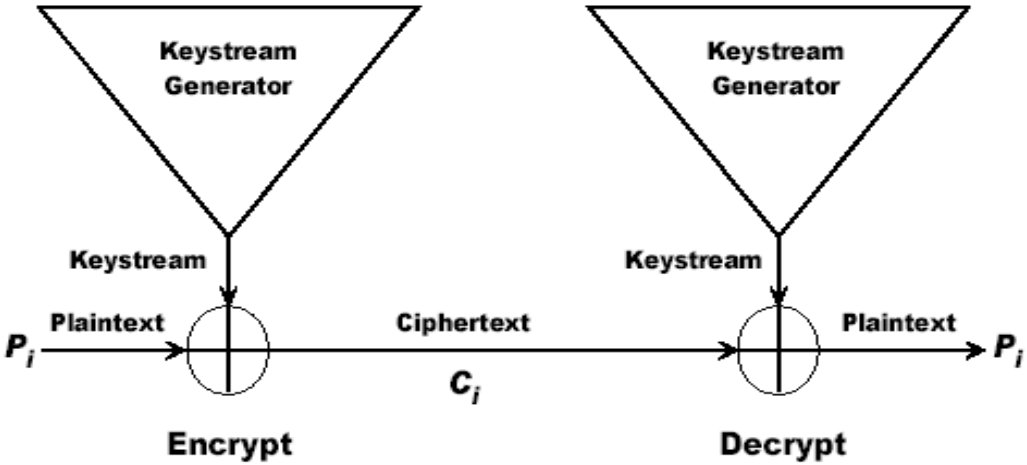
Mode de chiffrement avec contre-réaction du bloc de sortie:*Output-Feedback Mode*-OFB (5)

- Sécurité:
  - Vecteur d'initialisation unique et aléatoire.
- Propagation des erreurs:
  - Une erreur dans C_i affecte uniquement le bit correspondant de X_i .
  - Danger de perte de synchronisation. Il faut que les *shift registers* (registres de décalage) soient identiques lors du chiffrement et du déchiffrement.

Chiffrement par flux (*Stream Cipher*) (1)

- L'opération de chiffrement s'opère sur chaque élément du texte clair (caractère, bits).
- La structure d'un chiffrement par flux repose sur un générateur de clé qui produit une séquence de clés k1, k2,…,ki

$$c_i = m_i \oplus k_i$$
$$m_i = c_i \oplus k_i$$



Chiffrement par flux (*Stream Cipher*) (2)

- La sécurité du chiffrement dépend de la qualité du générateur :
  - si k_i =0 quelque soit i, M=C
  - si la séquence des clés k_i est infinie et complètement aléatoire, on obtient un One-Time-Pad.
  - En pratique, on utilise une séquence pseudo-aléatoire.
- Propagation des erreurs:
  - Une erreur dans C_i n'affecte qu'1 bit de M_i .
  - La perte ou l'ajout d'un bit de C_i affecte tous les bits suivants de M après déchiffrement.

Chiffrement par flux (*Stream Cipher*) (3)

- Types de chiffrement par flux :
  - *One-time-pad* : secret parfait (Shannon).
  - *Synchronous stream ciphers* : la clé est générée indépendamment du message clair/chiffré.
  - *Self-synchronizing* ou *asynchronous stream cipher* : la clé est générée en fonction de la clé et d'un nombre fixe de digits du texte chiffré précédemment.

PRNG (1)

- Générateur de nombres pseudo-aléatoires (PRNG)
  - Automate à nombre fini d'états qui à partir de la donnée d'un nombre fini de symboles (graine ou germe, *seed* en anglais) produit une suite potentiellement illimitée de symboles qui a l'apparence d'une suite aléatoire.
  - Si la graine est connue, de nombreux nombres sont générés rapidement et peuvent être reproduits plus tard.
- Formellement : un triplet formé par :
  - Un ensemble fini d'états.
  - Une fonction (déterministe) de transition qui transforme l'état de l'automate.
  - Une fonction (déterministe) de sortie qui associe un symbole à chaque état.

PRNG (2)

- Caractéristiques :
  - Efficace : peut produire de nombreux nombres en peu de temps.
  - Déterministe : une séquence de nombres peut être reproduite si le point de départ de la séquence est connu. Le déterminisme est pratique pour le déchiffrement.
  - Périodique : la séquence finira par se répéter (non souhaitable).
- Ex. : *Linear Congruential Generator* (le plus courant/ancien), défini par la relation de récurrence :

$$X_{n+1} = (aX_n + c) \bmod m$$

where X is the sequence of pseudo-random values

m, $0 < m$ - modulus

a, $0 < a < m$ - multiplier

c, $0 \leq c < m$ - increment

x_0, $0 \leq x_0 < m$ - the seed or start value

Linear feedback shift registers (LFSRs)

- Travail demandé :

1. Présentation, propriétés, utilisation en cryptographie.
2. Cryptanalyse des LFSR

Conclusion : problème de gestion de clés symétriques

- Le gestionnaire de clés doit avoir les caractéristiques suivantes :
  - capacité à gérer un grand nombre de clés,
  - gestion des clés de manière centralisée,
  - possibilité de gérer un renouvellement fréquent et automatique des clés,
  - sécurisation du serveur, seul garant du secret des clés.
- Problème de l'initialisation : comment réussir à utiliser, sans compromission, la 1ère clé pour établir le dialogue.
- Les principales implémentations industrielles du chiffrement à clés secrètes concernent :
  - les boîtiers de chiffrements,
  - les systèmes d'authentification du type Kerberos,
  - Autres applications répondant à des besoins spécifiques.